Aalto University
School of Science
Degree Programme of Computer Science and Engineering

Eric Malmi

Human Mobility Prediction:
A Probabilistic Transfer Learning Approach

Master's Thesis
Espoo, February 26, 2013

Supervisor: Prof. Erkki Oja
Instructor: Daniel Gatica-Perez, Ph.D.

Aalto University
School of Science
Degree Programme of Computer Science and Engineering

ABSTRACT OF
MASTER'S THESIS

Author: Eric Malmi

Title:
Human Mobility Prediction: A Probabilistic Transfer Learning Approach

Date: February 26, 2013 **Pages:** viii + 71

Professorship: Information and Computer Science **Code:** T-61

Supervisor: Prof. Erkki Oja

Instructor: Daniel Gatica-Perez, Ph.D.

Human mobility exhibits various daily and weekly routines, such as a home–work–lunch–work pattern which many working individuals follow regularly. In this thesis, a probabilistic method for predicting these mobility patterns is developed. Mobility prediction has a wide range of applications from context-aware smartphone applications to the modeling of epidemic disease spreading.

We use two sources of location data: the Mobile Data Challenge (MDC) data set which contains visit sequences recorded automatically through GPS and Wi-Fi sensors and the Foursquare (4sq) data set which consists of manual check-ins people have made in places such as train stations and restaurants. Our first goal is to study how the difference in the nature of the two location disclosure systems is reflected in mobility patterns. Differences related to, e.g., the number of check-ins are identified but the time distributions of the visits/check-ins turn out to be similar, suggesting that the two data sets can be used to complement each other.

The second goal is to develop a probabilistic next place prediction method. A model combining the strengths of the current state-of-the-art methods is derived and we show that it outperforms the current methods. Furthermore, the developed method is compatible with 4sq data allowing transfer learning.

The final goal is to introduce a transfer learning method in order to use 4sq data to complement the MDC data set. The method we propose is based on mixtures of multinomials and we show that it improves next place prediction accuracy during the first month of the data collection. Thus the proposed transfer learning method helps to tackle the cold start problem that many applications requiring the estimation of probability distributions face.

Keywords: human mobility, prediction, transfer learning, probabilistic methods, mixture of multinomials

Language: English

Aalto-yliopisto
Perustieteiden korkeakoulu

Aalto-yliopisto
Perustieteiden korkeakoulu
Tietotekniikan tutkinto-ohjelma

DIPLOMITYÖN
TIIVISTELMÄ

Tekijä:	Eric Malmi		
Työn nimi:	Probabilistinen siirto-oppimismenetelmä ihmisten liikkumisen ennustamiseksi		
Päiväys:	26. helmikuuta 2013	**Sivumäärä:**	viii + 71
Professuuri:	Tietojenkäsittelytiede	**Koodi:**	T-61
Valvoja:	Prof. Erkki Oja		
Ohjaaja:	Daniel Gatica-Perez, Ph.D.		

Ihmisten liikkumisessa on havaittavissa useita päivä- ja viikkorytmejä kuten koti–työ–lounas–työ-rytmi, joka on tyypillinen monille työssäkäyville henkilöille. Tässä työssä kehitetään probabilistinen menetelmä ihmisten liikkumisen ennustamiseksi. Liikkumisen ennustamisella on useita sovelluksia alkaen kontekstitietoisista matkapuhelinsovelluksista aina epidemioiden leviämisen mallintamiseen.

Työssä käytetään kahta paikkatietoaineistoa: Mobile Data Challenge (MDC) -aineistoa, joka sisältää GPS- ja Wi-Fi-sensoreiden avulla automaattisesti kerättyjä vierailusekvenssejä, sekä Foursquare (4sq) -aineistoa, joka koostuu manuaalisesti kirjatuista vierailuista eri paikkoihin kuten juna-asemiin ja ravintoloihin. Työn ensimmäisenä tavoitteena on tarkastella, miten näiden kahden paikkatiedonkeruumenetelmän erilaisuus näkyy aineistoista löytyvissä liikkumisrytmeissä. Osoittautuu, että eroavaisuuksia löytyy muun muassa tallennettujen vierailujen lukumäärissä, mutta toisaalta vierailujen aikajakaumat ovat samankaltaisia. Tämän perusteella voidaan päätellä, että aineistoja voidaan käyttää täydentämään toisiaan.

Työn toisena tavoitteena on kehittää probabilistinen menetelmä henkilön seuraavan sijainnin ennustamiseen. Johdettu menetelmä perustuu tämän hetken parhaisiin menetelmiin, ja työssä osoitetaan, että menetelmä suoriutuu paremmin kuin nykyiset menetelmät. Lisäksi menetelmä on yhteensopiva 4sq-aineiston kanssa, mikä mahdollistaa siirto-oppimisen.

Työn kolmentena tavoitteena on kehittää siirto-oppimismenetelmä, joka käyttää 4sq-aineistoa täydentämään MDC-aineistoa. Työssä osoitetaan multinomiaalimikstuurimalleihin perustuvan menetelmän parantavan seuraavan sijainnin ennustustarkkuutta, kun aineistoa on kerättynä alle kuukauden ajalta. Näin ollen menetelmä auttaa ongelmassa, joka kohdataan lukuisissa sovelluksissa, joissa vaaditaan todennäköisyysjakaumien estimointia, mutta joissa aineistoa ei ole aluksi riittävästi.

Asiasanat:	ihmisten liikkuminen, ennustaminen, siirto-oppiminen, probabilistiset menetelmät, multinomiaalimikstuurimalli
Kieli:	Englanti

Acknowledgements

This work was carried out in the Social Computing group at the Idiap Research Institute in Switzerland and it was part of the LS-CONTEXT project funded by Nokia. I would like to thank the leader of the group, my instructor Daniel Gatica-Perez. His insightful comments always helped me overcome the challenges I faced. I also want to thank Erkki Oja for supervising the thesis. At Idiap, it was a pleasure to have the opportunity to work with Trinh Minh Tri Do who would always come up with new ideas to try out. I also enjoyed bouncing my ideas off of Darshan and other friends at Idiap.

I was originally introduced to the Nokia data set by Juha Raitio. We explored the data set together and with others including Arno Solin, Timo Honkela, Oskar Kohonen, and Krista Lagus. These studies set the stage for my thesis project. I also had many useful discussions with Jan Blom from the Nokia Research Center.

Much of what I have learnt during my university studies have come not only from lecturers but also from other students whom I have collaborated with. Regarding the thesis, I got very useful comments on my text from Arno, Atle, Jussi, Marko, Matti, and Mikael.

I want to thank my parents, my two sisters, and Maria for their constant support and encouragement throughout the thesis project.

Finally, I wish to express my gratitude to the One who is above all:

"Does he not see my ways and count my every step?" (Job 31:4)

Ryttylä, February 26, 2013

Eric Malmi

Contents

Symbols and Abbreviations

Symbols

x	Place, location
t	Check-in time
t^e	End time of a visit
t^s	Start time of a visit
h	Hour $\in \{0, 1, \ldots, 23\}$
d	Weekday $\in \{1, 2, \ldots, 7\}$
w	Weekend $\in \{0, 1\}$
$\mathbf{t}$	Check-in time vector of a place
$\mathbf{T}$	Set of check-in time vectors of all places
$\mathcal{H}_{li}$	Number of check-ins to place l occurring at hour i
$\mathcal{D}_{lj}$	Number of check-ins to place l occurring on weekday j
$\mathbb{E}_z[f(x, z)]$	Expectation of function $f(x, z)$ with respect to random variable z
$\mathcal{L}$	Log likelihood function
$\boldsymbol{\Theta}$	All model parameters
$\boldsymbol{\theta}$	Parameters of a multinomial hour distribution
$\boldsymbol{\varphi}$	Parameters of a multinomial weekday distribution
z	Hidden variable
L	Number of places
C	Number of mixture components
N	Number of check-ins
N_l	Number of check-ins to place l
N_p	Number of user's previous visits to a single MDC place
N_u	Number of user's previous place transition
N_a	Number of user's previous place transition available for training
π	Mixing coefficient in Method II
Δt_n	Travel time between visits n and $n + 1$

Δx_n	Travel distance between visits n and $n+1$
α	Transfer learning coefficient

Abbreviations

4sq	Foursquare
BT	Bluetooth
Cell ID	Identity of a cell in a GSM network
Combined	Multinomial check-in time model estimated both from MDC and 4sq data
EM	Expectation-Maximization
GPS	Global Positioning System
HPY	Hierarchical Bayesian n–gram model based on Pitman–Yor processes
KDE	Kernel density estimation
LBSN	Location-based social network
LDCC	Lausanne Data Collection Campaign
M1	Method I
M2	Method II
MDC	Mobile Data Challenge
MDC only	Multinomial check-in time model estimated only from MDC data
MLE	Maximum likelihood estimate
PDF	Probability density function
PPD	Posterior predictive distribution
vpd	Number of visits per day
Wi-Fi	Used as a synonym for WLAN (wireless local area network)

1

Introduction

Human behavior, as arbitrary as it sometimes may seem, exhibits various routines and a certain degree of predictability [35]. Considering human mobility patterns, in particular, it is easy to find, e.g., a *home-work-lunch-work* pattern in the lives of many working individuals and therefore, it is safe to assume that mathematical models describing human mobility patterns can be discovered.

In addition to being an interesting research question in itself, human mobility prediction has also several potential applications. The rise of smartphone users has opened the doors for context-aware applications which utilize the sensing capabilities of the phones in order to provide users with content that is customized for their current or future location. For instance, if your smartphone can infer that you are probably going to have a dinner next, it may recommend you the most popular restaurant nearby and the fastest route to reach it. Furthermore, understanding mobility patterns can help in designing public transport systems or studying the spread of viral diseases [10].

Location data used in mobility studies originates from two different types of systems, namely *manual* and *automatic check-in systems*. In the former, users manually disclose their location through location-based social networks (LBSNs) such as Foursquare (4sq) [6, 25, 36], while in the latter ones, a smartphone records the location traces of a user through mobile sensors [17, 18, 23, 41]. Manual check-in systems are driven by individual preferences, specific incentives, and bounded by people's interest and attention, which has been shown in a few studies [6, 25]. These studies have investigated what motivates LBSN users to check in to certain places and how people's self-representation choices affect location sharing decisions. For instance, Lindqvist et al. [25] found that some users do not check in at places they consider boring, such as their home, while oth-

ers do not share their visits to fast food restaurants as they consider those embarrassing. In contrast to the subjective nature of manual LBSN check-in patterns, automatic check-in systems (in which the location is inferred, e.g., from GPS trajectories and wireless access points) are agnostic to the above issues and provide objective information about users' whereabouts but often suffer from their own limitations including sensor failures and battery constraints. One of the goals of this thesis is to study how the overall behavior and mobility patterns of 4sq users, compared with the behavior and patterns of MDC users, reflect the different nature of manual and automatic check-ins.

Our automatic check-in data comes from the *Nokia Mobile Data Challenge* (MDC) [24], which in turn originates from the Lausanne Data Collection Campaign (LDCC) [20]. The MDC data set contains daily life data from 80 users and about 16 months. The users were given a smartphone which used a variety of sensor data to infer instantaneous locations and visited places of the users. The Mobile Data Challenge had three dedicated tasks, one of which was the *Next Place Prediction Task*. In this task, the objective was to predict the next visit location of a user given various sensor information about the user's current visit and some previous visits. Each team reported their approach and the papers of the three best teams, in terms of prediction accuracy, were published. We aim to improve these state-of-the-art methods by deriving a probabilistically sound model which combines the strengths of the individual approaches. Each of the top entries considers only the place and the end time of the current visit and thus discards all the other information about the visits such as Bluetooth or Wi-Fi observations. In the same manner, we limit ourselves to studying only the spatio-temporal properties of visits. Inclusion of additional contextual information was studied by the winning MDC team but no improvements were observed [13].

As an instance of a manual check-in system, we use data from 4sq. This data is collected from Switzerland, where also the MDC data has been collected. Our focus is in predicting the mobility patterns of MDC users, and we want to improve these predictions by including additional information from the 4sq data set. The user bases of the two data sets are probably mostly separate, but we show that we can match places or place categories across the data sets. More specifically, we show how to improve the modeling of the temporal characteristics of a MDC place by incorporating prior knowledge of place visit time distributions from 4sq. Due to the aforementioned differences in the two data sets, the 4sq data is from a different but a related domain compared to the MDC data, which is a typical transfer learning problem setting [33]. To our knowledge, this is the first attempt

to use mobility data from different systems in a complementary manner.

Gao et al. [14] draw a compelling parallel between location sequences and text documents; a set of check-ins of a person or a place is comparable to a text document whose each word corresponds to a check-in and whose length can vary. This insight provides a whole arsenal of methods used in natural language processing (described extensively, e.g., in [27]) for studying location sequences. The approach we take for transfer learning involves clustering the places and learning the check-in time distribution of each cluster. In the context of text documents, the clustering problem can be solved using *mixtures of multinomials* [34], and thus we adopt the same approach for place clustering.

To summarize, the three main goals of this thesis are:

1. Study the differences and similarities of manual check-ins from 4sq and automatically inferred visits from MDC.

2. Derive a probabilistic next place prediction model which is applicable to transfer learning and based on the current state-of-the-art approaches.

3. Introduce a transfer learning method for learning the visit time distributions of places using two data sets.

The rest of the thesis is organized as follows. Chapter 2 discusses related work. Probability distributions and estimation techniques that are used throughout the thesis are presented in Chapter 3, and a comparison of the two data sets we use is conducted in Chapter 4. The next place prediction problem and the results related to it are discussed in Chapter 5. In Chapter 6, we introduce a method for learning time distributions across the data sets and apply it to the next place prediction problem. Finally, we present discussion and draw conclusions in Chapter 7.

2

Related Work

Automatic check-in data can be obtained from various sources. Cell ID information is one of the earliest and most used automatic sources for studies on human mobility prediction [2, 7]. This positioning system is based on obtaining information about nearby cell towers whose coordinates are known, and it usually results in rather coarse-grained location information, depending on the density of the cell towers. Cell ID data is still in active use. For example, Gonzalez et al. [16] applied methods from statistical physics to the next place prediction problem and Song et al. [35] used entropy to measure the limits of predictability based on cell ID data. A widely analyzed data set containing information about human mobility also based on cell ID information is the *reality mining* data set [11] for which the next place prediction problem was studied by Eagle and Pentland [12]. In their work, next place prediction was handled as a missing value imputation problem which was solved using the eigendecomposition of a location data matrix. Nowadays, smartphone sensors, such as GPS, provide accurate location data. However, instead of raw GPS trajectories, it is more convenient to study visit sequences between places which can be extracted from GPS traces and/or other data types such as Wi-Fi/GSM radio. In the place learning literature, there are several definitions of place, such as a small circular region [18], Wi-Fi/GSM fingerprint [17, 19, 23], or a multivariate Normal distribution [32]. Recently, Zheng et al. [41] proposed a two-stage algorithm that first detects stay points from GPS traces and then clusters these into places. The algorithm was later improved by Montoliu and Gatica-Perez [29] by taking into account other sensors such as Wi-Fi. In the MDC data, the visit sequences have been extracted using the improved version [24].

Recently, manual check-in data from location sharing services, including Foursquare, Whrrl, and Gowalla, have become a popular source for

studies on human mobility since they provide more accurate information about users' whereabouts than cell ID data. Gao et al. [14] applied methods used in statistical natural language modeling for predicting the next check-in of a user while the focus of the work by Noulas et al. [31] was in extracting a large set of user specific features and global features for the prediction task. Instead of using prediction accuracy as the performance measure like most of the studies on next place prediction, the latter work measured the performance of the predictions by *average percentile rank* which is typically used in information retrieval tasks. Cho et al. [5] introduced a probabilistic model for predicting the location of a user given only the time context. Their model was designed to take into account the finding that long-distance travel is more influenced by the social network ties of the user, and the performance of the method was assessed on three different check-in data sets. General mobility patterns found in manual check-in data sets were studied by Cheng et al. [4] who collected a data set of 22 million manual check-ins mainly from Foursquare but from several other applications as well. One thing they looked into was the distribution of displacement distances. Later on, Noulas et al. [30] showed that rather than the distance of a displacement, it is the rank of the displacement what characterizes its probability more accurately. Another aspect Cheng et al. studied was the daily and weekly check-in patterns of the users. Ye et al. [40] looked into this type of patterns in Whrrl data, but in addition they analyzed the daily and weekly check-in patterns of different types of places, showing that places can be characterized not only by their user-assigned category tags, but also by their temporal check-in patterns.

The reasons for using location sharing services have been studied as well. Lindqvist et al. [25] conducted interviews and two surveys in order to understand what motivates people to use Foursquare to share their location. They reported several reasons why people use it including a gaming aspect, keeping in touch with friends, and discovering new places. Cramer et al. [6] also conducted interviews and a survey in order to gain insight into Foursquare usage and they were able to confirm the findings by Lindqvist et al. Additionally, they looked into the demographics of people's Foursquare friends.

Much work has been done to study automatic and manual location disclosure systems separately. To our knowledge, the only work that has used both automatic and manual check-in data is by Cho et al. [5] and the objective of their study is to look at specific mobility patterns and see if those can be found in different datasets. In contrast, one of the goals of this thesis is not only to test our models on two different data sets but to use the two data sets (MDC and 4sq) to complement each other. Learning from mul-

tiple sources is currently an active research topic in the field of machine learning, and typical transfer learning problems include different types of classification and regression tasks [33]. In our case, we want to do density estimation using additional data from a source domain (4sq) that does not follow exactly the same distribution as our target domain (MDC) but is related. This is an *unsupervised transfer learning* problem which is a relatively new and little studied research topic [33]. While different sources of check-ins have not been used to complement each other previously, the check-ins of different users have been used to enhance the next place prediction for a single user [8, 14, 31].

Finally, the top three methods in the MDC Next Place Prediction Task were by Etter et al. [13] (accuracy 56.2 %), Wang et al. [39] (accuracy 52.8 %), and Gao et al. [15] (accuracy 52.4 %). The winning method used an ensemble of a probabilistic classifier, artificial neural network, and decision trees. Ensemble methods, such as this, have proved to produce excellent classification results, e.g., in the Netflix competition [21, 38]. In this work, we consider only probabilistic classifiers and thus limit ourselves to study the probabilistic classifier by Etter et al. In addition, their method implemented an aging factor and took into account if a person changed their residential location during the data collection period. However, the actual implementation of these improvements was not discussed in their paper, and therefore, we do not implement them. The second MDC team compared a probabilistic model and support vector machines, which consider next place prediction as a multiclass classification task, but did not combine the two approaches. The third team had a very similar probabilistic model than the second team and since the third team presented their method in more detail, we follow the model formulation of their paper [15] in the latter parts of this work. What the top three methods had in common is that they all contained a spatial distribution, namely a Markov model, and a time distribution, capturing the temporal characteristics of different places.

3

Mathematical Background

Certain formulas related to probability distributions and techniques used to estimate them occur frequently throughout the thesis. In this chapter, we go through these basic formulas and techniques.

3.1 Probability Theory

One of the most fundamental theorems in probability theory is the *Bayes' theorem* which serves as the basis for several probabilistic techniques [3]. The theorem states that the conditional probability of two random variables A and B is given by

$$p(A = a \mid B = b) = \frac{p(A = a)p(B = b \mid A = a)}{p(B = b)}.$$

For notational convenience, we simply write

$$p(A \mid B) = \frac{p(A)p(B \mid A)}{p(B)}.$$

Sometimes, we are only interested in the most probable value of A given B, in which case we can ignore the marginal probability in the denominator and get

$$p(A \mid B) \propto p(A)p(B \mid A).$$

However, if we need to consider the marginal probability, it is usually extended using *marginalization*

$$p(B) = \sum_A p(B, A) = \sum_A p(B \mid A)p(A),$$

where the summation is over all possible values of A.

3.2 Maximum Likelihood Estimation

Let us have data samples $\mathbf{Y} = [y_1, y_2, \ldots, y_N]$, and assume that the samples follow distribution $p(y \mid \boldsymbol{\Theta})$, where $\boldsymbol{\Theta}$ are the parameters of the distribution. Our task is to find parameter values that maximize the probability of the observed data $\mathbf{Y}$.

We assume that the samples are independent and identically distributed, and thus the *likelihood function* takes the form

$$\ell(\boldsymbol{\Theta} \mid \mathbf{Y}) \equiv p(\mathbf{Y} \mid \boldsymbol{\Theta}) = \prod_{i=1}^{N} p(y_i \mid \boldsymbol{\Theta}).$$

Now we want to find $\boldsymbol{\Theta}$ that maximizes this expression and typically this is easier to achieve if we work with the log likelihood. Taking the logarithm does not change the maximum since logarithm is a monotonous function

$$\mathcal{L}(\boldsymbol{\Theta} \mid \mathbf{Y}) = \sum_{i=1}^{N} \log p(y_i \mid \boldsymbol{\Theta}).$$

In a simple case, the value of $\boldsymbol{\Theta}$ that maximizes this expression can be found by setting the derivative of the expression to zero with respect to $\boldsymbol{\Theta}$. This value is called the *maximum likelihood estimate* (MLE).

3.3 Expectation-Maximization Algorithm

In case we have latent variables or missing data, it can be challenging to find the maximum likelihood estimates of the parameters. In these cases, the *Expectation-Maximization* (EM) algorithm can be used to solve the MLEs iteratively [3, 9, 28].

Let us denote the observed data by $\mathbf{Y}$, the latent variables by $\mathbf{Z}$, and the model parameters by $\boldsymbol{\Theta}$. The observed data is called the *incomplete-data* and the observed data together with the latent variables form the *complete-data* set whose log likelihood function is denoted $\log p(\mathbf{Y}, \mathbf{Z} \mid \boldsymbol{\Theta})$. The EM algorithm starts by initializing $\boldsymbol{\Theta}^{\text{old}}$, e.g., randomly. In the *expectation* (E) step, we evaluate $\mathcal{Q}(\boldsymbol{\Theta}, \boldsymbol{\Theta}^{\text{old}})$, which is the expectation of the complete-data log likelihood with respect to the latent variables

$$\begin{aligned} \mathcal{Q}(\boldsymbol{\Theta}, \boldsymbol{\Theta}^{\text{old}}) &= \mathbb{E}_{\mathbf{Z}}[\log p(\mathbf{Y}, \mathbf{Z} \mid \boldsymbol{\Theta})] \\ &= \sum_{\mathbf{Z}} p(\mathbf{Z} \mid \mathbf{Y}, \boldsymbol{\Theta}^{\text{old}}) \log p(\mathbf{Y}, \mathbf{Z} \mid \boldsymbol{\Theta}). \end{aligned} \tag{3.1}$$

In the *maximization* (M) step, we calculate new parameter values $\boldsymbol{\Theta}^{\text{new}}$ by maximizing expression $\mathcal{Q}$ as follows

$$\boldsymbol{\Theta}^{\text{new}} = \arg\max_{\boldsymbol{\Theta}} \mathcal{Q}(\boldsymbol{\Theta}, \boldsymbol{\Theta}^{\text{old}}). \tag{3.2}$$

After this, we update $\boldsymbol{\Theta}^{\text{old}} \leftarrow \boldsymbol{\Theta}^{\text{new}}$ and go back to the E step.

It can be shown that the iterations of the EM algorithm never decrease the incomplete-data likelihood which we want to maximize [9]. However, the iteration may converge to a local optimum instead of the global one, and therefore, the EM algorithm is typically run several times with different initializations of $\boldsymbol{\Theta}^{\text{old}}$. The run which has yielded the highest likelihood is selected.

The algorithm is further illustrated in Section 6.2.2 where we use it to learn a mixture model and in Appendix A where we derive the EM equations for the mixture model.

3.4 Time Distribution Estimation

The estimation of check-in time distributions plays a major role in this work. In this section, we present three techniques for estimating these distributions. We define that a check-in time t consists of the hour of the day h and the day of the week d. Furthermore, we make a simplifying assumption that these two are independent, i.e. $p(t) = p(h, d) = p(h)p(d)$.

3.4.1 Multinomial Distribution

Let us assume that hour h and weekday d are categorical variables. That is, we define

$$h \in \{0, 1, \ldots, 23\}$$
$$d \in \{1, 2, \ldots, 7\}.$$

Multinomial distribution is a natural way to model this kind of data.

Let $\mathbf{h}$ be the hours of N check-ins. We transform this into matrix $\mathbf{Y}$ which is the 1-of-K encoding of the hour vector, where $y_{ni} = 1$ if $h_n = i$ and zero otherwise. The probabilities of different hours are defined by parameters $\boldsymbol{\theta}$, where θ_i is the probability of hour i. The probability distribution of a single check-in hour is thus

$$p(\mathbf{y}_n \mid \boldsymbol{\theta}) = \prod_{i=0}^{23} \theta_k^{y_{ni}}.$$

Let $\mathcal{H}_i = \sum_{n=1}^{N} y_{ni}$ be the count representation of the check-in hours. Now we obtain the distribution of N check-in hours in the form

$$p(\mathbf{Y} \mid \boldsymbol{\theta}) = p(\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N \mid \boldsymbol{\theta}) = \prod_{n=1}^{N} \prod_{i=0}^{23} \theta_i^{y_{ni}} = \prod_{i=0}^{23} \theta_i^{\mathcal{H}_i}.$$

We can also calculate the distribution for a count vector $\mathcal{H} = [\mathcal{H}_0, \mathcal{H}_1, \ldots, \mathcal{H}_{23}]$ by

$$p(\mathcal{H} \mid \boldsymbol{\theta}) = \frac{N!}{\prod_{i=0}^{23} \mathcal{H}_i!} \prod_{i=0}^{23} \theta_i^{\mathcal{H}_i}.$$

This is called the *multinomial distribution*. For the count vector it holds that $N = \sum_{i=0}^{23} \mathcal{H}_i$.

It can be shown (see e.g. [3]) that the MLE for the parameters is

$$\theta_i = \frac{\mathcal{H}_i}{N}. \tag{3.3}$$

However, if N is small, we typically get a lot of zero probabilities which is undesirable if we want to calculate likelihoods. In order to avoid this problem, we apply *Laplace smoothing* [27], which assigns a *pseudocount* of one to each category. The resulting parameter estimates are

$$\theta_i = \frac{1 + \mathcal{H}_i}{24 + N}. \tag{3.4}$$

Note that $\sum_{i=0}^{23} \theta_i = 1$.

An example of a multinomial hour distribution based on the actual check-in hours of a MDC place is shown in Figure 3.1. The distribution is never zero due to the smoothing.

In a similar fashion, we can model weekdays, but instead of 24 categories, we have 7 categories. If $\mathcal{D}$ is the weekday count vector, where $\mathcal{D}_i$ is the number of check-ins having occurred on day i (Mon $= 1$, Tue $= 2, \ldots,$ Sun $= 7$) and $\boldsymbol{\varphi}$ are the weekday probabilities, the distribution of a check-in time vector $\mathbf{t}$ is given by

$$\begin{aligned} p(\mathbf{t} \mid \boldsymbol{\theta}, \boldsymbol{\varphi}) &= p(\mathcal{H}, \mathcal{D} \mid \boldsymbol{\theta}, \boldsymbol{\varphi}) \\ &= p(\mathcal{H} \mid \boldsymbol{\theta}) p(\mathcal{D} \mid \boldsymbol{\varphi}) \\ &= \frac{N!}{\prod_{i=0}^{23} \mathcal{H}_i!} \left(\prod_{i=0}^{23} \theta_i^{\mathcal{H}_i} \right) \frac{N!}{\prod_{j=1}^{7} \mathcal{D}_j!} \left(\prod_{j=1}^{7} \varphi_j^{\mathcal{D}_j} \right). \end{aligned} \tag{3.5}$$

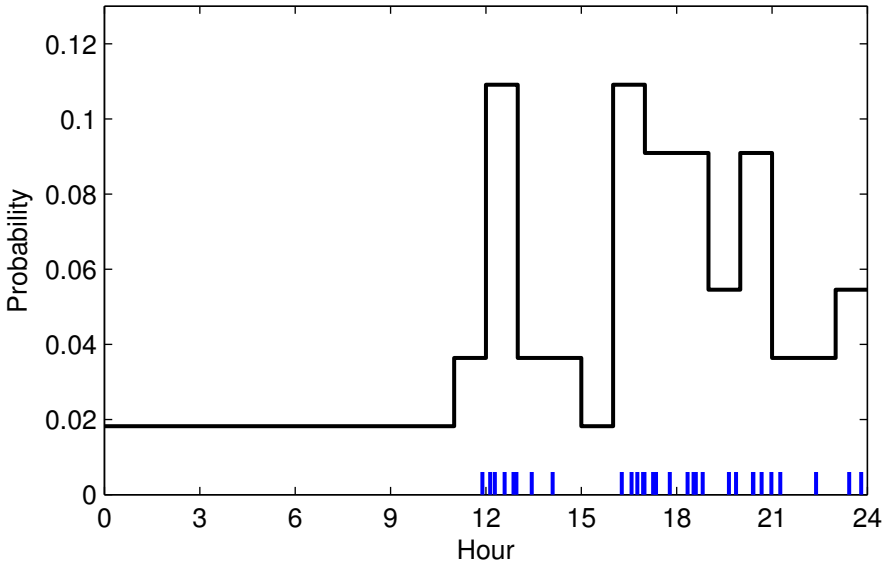

Figure 3.1: An example of a multinomial distribution with Laplace smoothing estimated based on the check-in times (shown with blue lines on the bottom) of a single place.

3.4.2 Gaussian Distribution

Another approach to modeling individual check-in hours is to consider them as continuous variables and estimate their *probability density function* (PDF). Gao et al. [15] suggest modeling the hours (h) and weekdays with a Gaussian distribution, which is a parametric continuous density estimate

$$h \sim \mathcal{N}(\mu, \sigma^2),$$

where μ is the mean of the distribution and σ^2 is the variance.

The distribution of N visit hours is given by

$$\begin{aligned}\log p(\mathbf{h} \mid \mu, \sigma^2) &= \log\left(\prod_{n=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\frac{(h_n-\mu)^2}{\sigma^2}}\right) \\ &= -\frac{N}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{n=1}^{N}(h_n-\mu)^2\end{aligned}$$

Setting the partial derivatives of this expression to zero with respect to μ

and σ^2 yields the following MLEs

$$\hat{\mu} = \frac{1}{N}\sum_{n=1}^{N} h_n \tag{3.6}$$

$$\hat{\sigma}^2 = \frac{1}{N}\sum_{n=1}^{N}(h_n - \hat{\mu})^2. \tag{3.7}$$

This distribution is applicable to continuous check-in times but Gao et al. seem to consider only integer check-in hours and days. Therefore, we also discretize the Gaussian into 24 bins, in case of the hour distributions, and into 7 bins, in case of the weekday distributions. Then we normalize the bins to sum up to one. This is illustrated in Figure 3.2, which shows an example of a Gaussian estimate and its discrete correspondent.

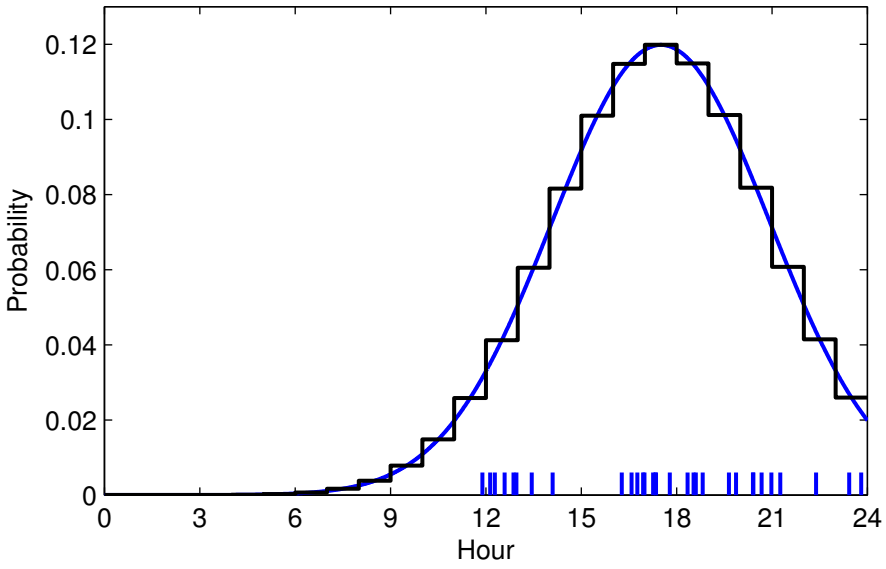

Figure 3.2: An example of a Gaussian distribution and its discretized version estimated from the check-in times (shown with blue lines on the bottom) of a single place.

3.4.3 Kernel Density Estimation

Finally, we present a nonparametric density estimation method called *kernel density estimation* (KDE) [3], which we use for modeling hour distributions but not the weekday distributions. Instead of assuming a certain

type of distribution and learning its parameters, KDE stores each training data point and applies a kernel function $K(\cdot)$ to these points.

The standard KDE function with N training data points is given by

$$p(h) = \frac{1}{Nw} \sum_{n=1}^{N} K\left(\frac{h - h_n}{w}\right), \tag{3.8}$$

where w is the window width parameter that controls the smoothness of the resulting density function and $K(\cdot)$ we define to be a Gaussian with variance of one hour. The main drawback of KDE is that it gets slow and memory consuming when the number of training data points increases [1]. To avoid this, to make it easier to normalize the density function, and to make the method comparable to the other two estimation methods, we discretize $p(h)$ into 24 bins each corresponding to a one hour time slot.

We make a minor modification to the distance metric of the standard KDE function in order to take into account the cyclic nature of h. In addition, we use Laplace smoothing in the estimation since it seems to provide better results than if we omitted it. Our final, unnormalized, KDE function takes the form

$$p(h) \propto 1 + \sum_{n=1}^{N} K\left(\min\{|h - h_n|,\, \min(h, h_n) + 24 - \max(h, h_n)\}\right). \tag{3.9}$$

These probabilities are scaled to sum up to one. The above expression calculates the minimum distance between hours h and h_n which are considered cyclic variables so that $h = 24$ is equivalent to $h = 0$. An example of a KDE estimate is displayed in Figure 3.3. Note that the distribution is continuous even at midnight.

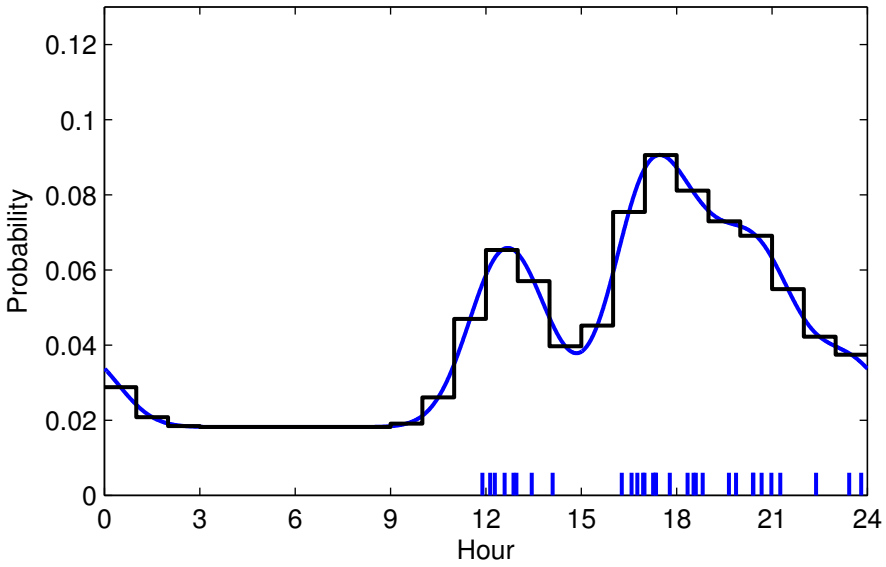

Figure 3.3: An example of a KDE distribution and its discretized version estimated based on the check-in times (shown with blue lines on the bottom) of a single place.

4

Data Set Comparison

In this chapter, we do a comparison of the two data sets that are used in this thesis. In Section 4.1, we present the MDC data set, which contains automatic check-in data, and in Section 4.2, we present the manual check-in data set from 4sq. In Section 4.3, we compare some general temporal and spatial patterns found in the data sets. Finally, in Section 4.4, we address the question of whether it is possible to find matching places across the two data sets. The motivation for this goal is that we want to use the data sets to complement each other. If we can accurately determine which 4sq place corresponds to a given MDC place, we may use this information, e.g., to infer the category of the MDC place or to gain some extra knowledge about when is the MDC place typically visited at. The work in this chapter was published in [26].

4.1 Mobile Data Challenge

The automatic check-in data comes from Nokia's Mobile Data Challenge[1], described in [24], which originates from the Lausanne Data Collection Campaign (LDCC) [20]. The MDC Dedicated Track data set contains daily life data from 80 users and about 16 months (Sep 2009 – Feb 2011). The population is concentrated in the Suisse Romande region, the French speaking part of Switzerland, but their data was recorded wherever they were inside the country. Users are a combination of students and professionals, mainly in the 22–33 age range.

The volunteers in LDCC were given smartphones that automatically recorded various types of sensory information, including GPS, Wi-Fi, Bluetooth, cell ID, and accelerometer data, and other types of information, such

[1] `http://research.nokia.com/page/12000`

as call logs and application usage logs. The MDC data set also contains place visit (= check-in[2]) sequences that were automatically inferred based on the GPS trajectories and Wi-Fi access points. In this work, we only use the place visit sequences. They contain the identifier labels of the visited places and the start and end times of the visits. A visit is defined to have a minimum duration of 20 minutes meaning that if a user stays in a place for less than 20 minutes, the user is considered being on the move. This implies that many actual places in daily life, involving short stays like bus stops, metro stations, etc. are likely not included in the data.

The algorithm for detecting the visited places is described in [29] and consists of two stages: In the first stage, the temporally consecutive location points of a user are grouped into *stay points*. Then in the second stage, the stay points are grouped into *stay regions*, i.e. the places. Each place has a pair of coordinates, which corresponds to the center of mass of the stay points, and a radius of 100 meters, which is the maximum distance from the center to the location points. The MDC organizers run the place detection algorithm for each user separately, so that each user has associated a unique set of places.

Some places have been given semantic labels by the users. The users have been asked to select labels for their most frequently visited places and some of the rarely visited places from a fixed list of category labels, such as home, work, restaurant, etc.

4.2 Foursquare

The manual check-in data comes from *Foursquare*[3] (4sq) which is a highly popular location sharing application. The data is collected through Twitter as some 4sq users have allowed their check-ins to be published on their Twitter stream, which enables the collection of longitudinal data per user. Similar techniques to collect 4sq data have been used in [4, 30]. For comparison purposes, we focus on the country-level check-ins from Switzerland, where the MDC data has also been collected.

The 4sq data has been collected between December 19, 2011 and June 21, 2012, and it contains 12882 different users in total. However, most of these users have only a couple of check-ins, possibly corresponding to an initial interest, and therefore we select *active users* who have made at least 40 check-ins and whose first and last visit are at least 4 weeks apart

[2]For notational convenience, *check-in* and *visit* are used interchangeably hereinafter.
[3]`https://foursquare.com/`

from each other. These choices result in 302 active users who are used for Section 4.3. A check-in consists of a latitude, longitude, place title, and for most places, a place category[4].

Check-ins from 4sq and MDC visits have three fundamental differences: The first and the main difference is that 4sq check-ins are manual. When a 4sq user wants to check in, the user gets a list of nearby places from which the user manually selects the suitable one (illustrated in Figure 4.1). Thus for most 4sq users the data is sparse as the users do not often check in at every place they go to. In addition, our 4sq data set has been collected through Twitter, and only a fraction of 4sq users share all or part of their check-ins this way. The second key difference relies on the definition of a place. While MDC places correspond to circular regions, 4sq places are physical coordinates suggested by the system which are more accurate in terms of precision. The third difference is that the MDC visits have both a start and an end time, whereas the 4sq visits have only the time of checking in.

Some basic statistics regarding both data sets are shown in Table 4.1 and the locations of all places are shown in Figure 4.2. While the 4sq places (red 'x') distribute rather evenly over the whole Switzerland, the MDC places (blue '+') are centered in the French-speaking part of Switzerland and especially around Lake Geneva where the MDC population lives. We have also included the places visited by inactive users. In total, there are 7281 MDC places and 17482 4sq places.

Table 4.1: Basic statistics of the MDC and 4sq data sets. *#labeled places* is the number of places for which the users have given a semantic label. *#active users* is the number of users with at least 40 visits and one month of data in total and *#visits* is the number of visits made by the active users.

	MDC	4sq
#places	7281	17482
#labeled places	398	16382
#users	80	12882
#active users	80	302
#visits	51607	40629
First visit	30 Sep 2009	19 Dec 2011
Last visit	4 Feb 2011	21 Jun 2012

[4]List of categories can be found at: `http://aboutfoursquare.com/foursquare-categories/`

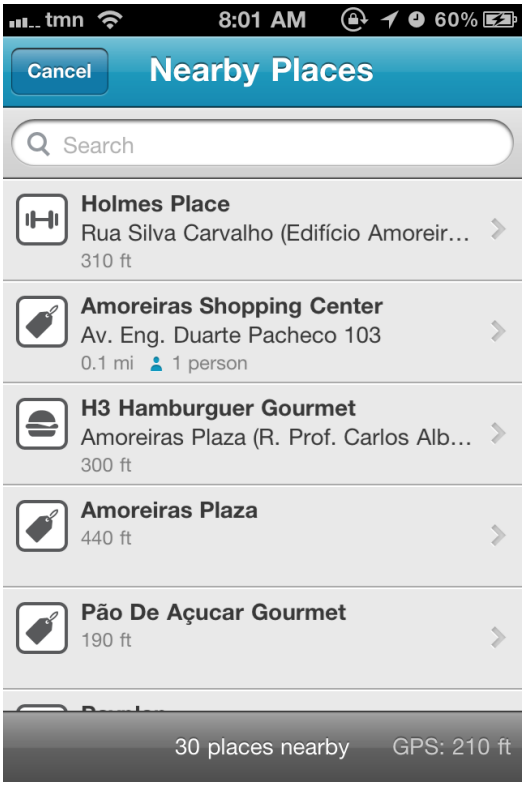

Figure 4.1: Screenshot from the Foursquare application where a user selects the place to check into from a list of nearby places. (The photo was uploaded by *Gustavo da Cunha Pimenta* and was taken from `http://www.flickr.com/photos/guspim/6236568269/`)

4.3 Visiting Behavior

This section focuses on comparing the visiting behavior in the MDC and 4sq data. In Sections 4.3.1 and 4.3.2, we examine two temporal features of visits, namely how often and when do people visit places, and in Section 4.3.3, we study a spatial aspect related to sequences of visited places.

4.3.1 How Often Do People Visit Places?

The distributions of the average number of visits per day (vpd) are shown in Figure 4.3, and they appear to be quite different for the two data sets. For most MDC users, vpd is between 2-4 (median = 3.1) and no user has a $vpd > 6$. On the other hand, more than half of the 4sq users have $vpd < 2$ (median = 1.7) but there is also one 4sq user who has as many as 19

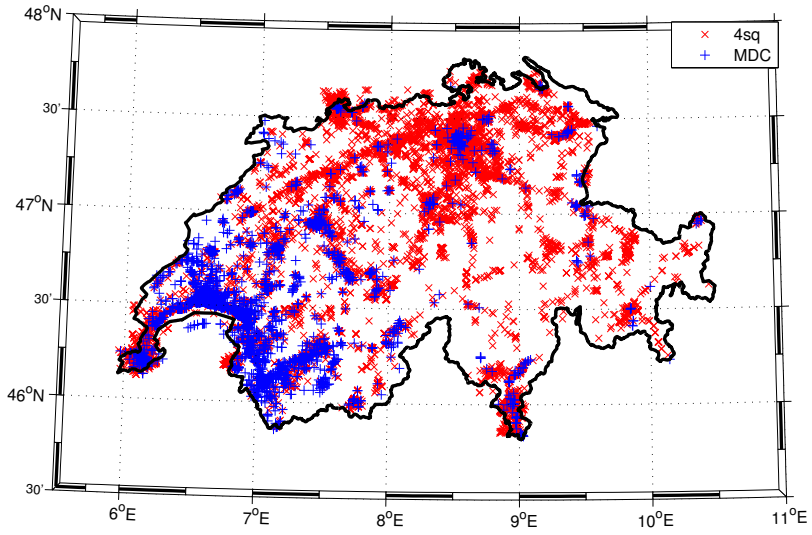

Figure 4.2: All MDC and 4sq places on a map of Switzerland. Note that in the Lake Geneva area, the two data sets are overlapped.

vpd. The two-tailed t-test with unequal variances at 99 % confidence level confirms that *vpd*s of MDC and 4sq users have different means. This shows

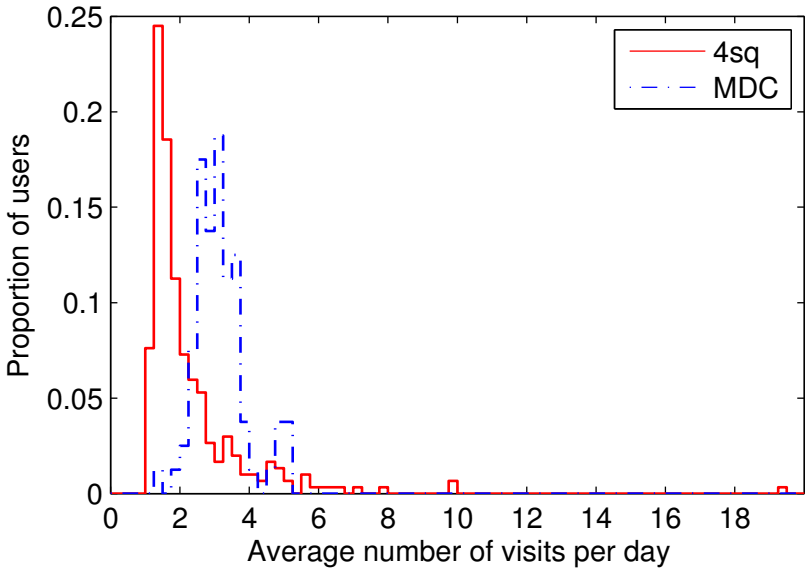

Figure 4.3: Empirical distributions of the average number of visits per day for manual and automatic systems.

an obvious advantage of automatic check-ins: many 4sq users check in

only occasionally, whereas MDC users record new visits regularly if they are carrying their mobile device with them. However, in the MDC data, a visit has been defined to last for at least 20 minutes. Thus, shorter visits will not be recorded, which might explain why the MDC data lacks a long tail, i.e., users with numerous visits per day which can be found in the 4sq data.

Let us also study if there are differences in the visiting behavior among the 4sq users. We define three user categories based on their average check-in activity: users with less than 1.5 visits per day (vpd) are denoted by "low", users with $1.5 \leq vpd \leq 4.5$ by "mid", and users with $vpd \geq 4.5$ by "high". In Table 4.2, we have calculated statistics of the weekly visit frequencies between the two data sets and between different 4sq user categories. The table shows how many visits people record per week, and how many distinct places and new places they record per week on average[5].

Table 4.2: Weekly averages for number of visits, distinct places, and new places visited during the week. Last column shows the number of users in each category. 4sq-* refer to activity categories of 4sq users with different amounts of visits per day.

	#visits	#distinct	#new	#users
MDC	17.0 ± 5.6	6.7 ± 2.0	3.2 ± 1.2	80
4sq	7.0 ± 7.6	4.7 ± 4.2	3.1 ± 2.6	302
4sq-low	2.7 ± 0.9	2.3 ± 0.7	1.7 ± 0.6	97
4sq-mid	7.0 ± 4.5	4.8 ± 3.0	3.3 ± 2.0	182
4sq-high	25.4 ± 13.3	13.9 ± 7.1	8.3 ± 4.4	23

MDC users record over twice as many (143 % more) visits per week and 43 % more distinct places than 4sq users but, quite interestingly, 4sq users record roughly the same number of new places. How to interpret these results? On one hand, the larger number of visits for the automatic case is a clear reflection of the fact that this system has no burden on human memory or attention compared to the manual case. But there are other reasons related to checking in at home or work. The MDC users who have labeled their home or workplace make on average 24 % and 17 % of their visits to these places, respectively. In contrast, a survey conducted by Lindqvist

[5]The rate at which people record new places decreases slightly as a function of time since the number of potential new places to explore decreases. In order to make the *new places per week on average* comparable between users with several months and users with only a couple of months, we reset the set of visited places to the empty set every two months.

et al. [25] showed that a vast majority of 4sq users never check in at their homes and less than half check in at their workplace on a daily basis. On the other hand, the fact that the overall number of new places are similar for both cases highlights the novelty-driven feature of location sharing systems, i.e., users are motivated to make the explicit effort of checking in, while for an automatic system, depending on its design, a new place might be treated just like previously seen places in terms of detection.

The numbers of distinct places and new places visited on average per week are shown for each user colored according to the user's activity category in Figure 4.4. There is clearly a correlation between these two variables which means that people who go to many distinct places per week also visit many new places per week. Both Table 4.2 and Figure 4.4 show that there are very different types of 4sq users in terms of activity levels.

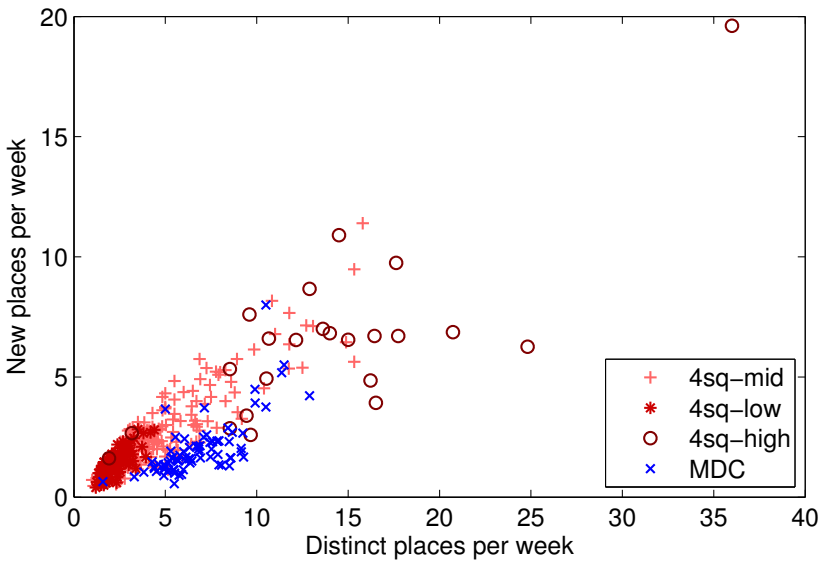

Figure 4.4: Each user's number of distinct and new places visited per week. The correlations between the two variables are 0.82, 0.93, 0.88, and 0.87 respectively for 4sq-low, 4sq-mid, 4sq-high, and MDC users.

4.3.2 When Do People Visit Places?

Figure 4.5 shows the distributions of MDC and 4sq visits over hours of the day and days of the week. The hour distributions have three peaks corresponding to the arrival to work between 8–10^{h}, lunch break around 12–14^{h} and some after-work activity between 18–19^{h}. Furthermore, MDC

visits peak in the night between 3–4^{h} which is due to a daily reset of the phone that causes artificial visits [20].

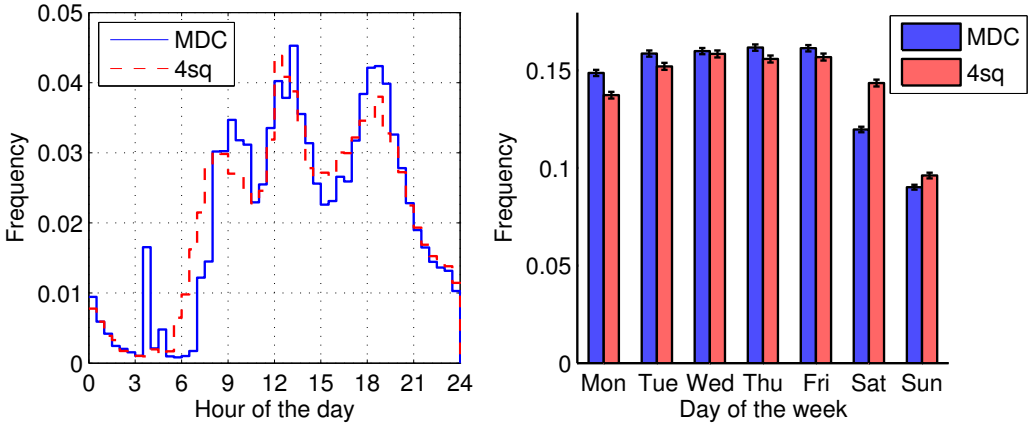

Figure 4.5: Distribution of visits over hours of the day (a) and days of the week with standard errors for the proportions (b).

Cheng et al. [4] have found the same pattern of three check-in peaks for other cities, namely Los Angeles, New York City, and Amsterdam. For these cities, the after-work peak is always the highest. Interestingly, in the 4sq and MDC data from Switzerland, the highest peak is around noon. This can be explained by cultural differences since in Switzerland the shops typically close earlier than many other countries (18:30^{h} during weekdays) and so urban activity decreases.

The weekday distributions on the right hand side of Figure 4.5 are also rather similar except for Saturday, when there are more 4sq visits than MDC visits. This interesting difference can be explained by looking at Figure 4.6, which shows the daily and weekly check-in time distributions for the 4sq user categories (see Sec. 4.3.1). The highly active users check in relatively less frequently in the morning than the *low* and *mid* users, and instead, they are more active in the afternoon. From the weekday distribution, we notice a difference in the behavior of the *low* users and the *high* users: the *high* users check in mainly on working days, especially on Wednesdays and Thursdays, whereas for the *low* users, Saturday is the most popular day for checking in. Keeping track of visited places is one of the reasons why people use 4sq [25] so the least active users probably check in mainly when they visit new places, instead of trying to win mayorships. On Saturdays, people are typically free to explore new urban places which can explain the popularity of Saturday among the least active 4sq users. Finally, Sunday is overall the least checked-in day for

both systems, which is not surprising given that shops and restaurants are mostly closed, except in touristic areas.

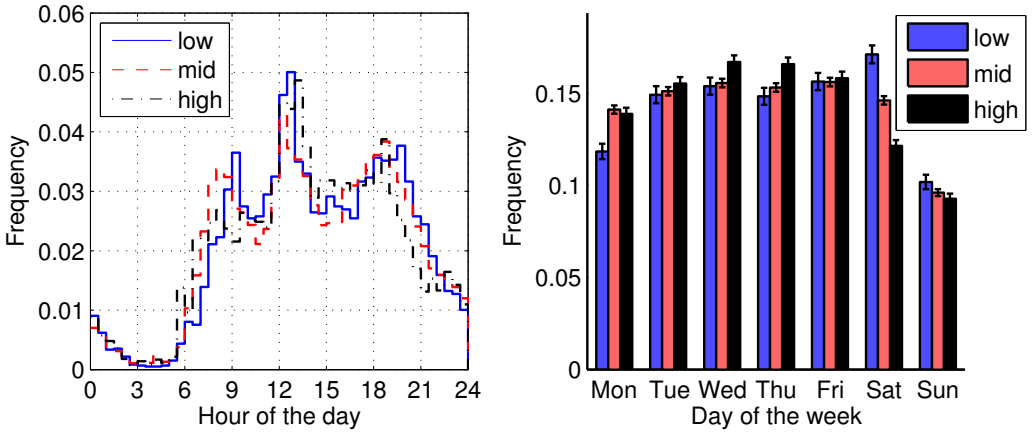

Figure 4.6: Visit time distributions for three 4sq user categories defined by their average check-in activity. low: users with $vpd < 1.5$; mid: users with $1.5 \leq vpd \leq 4.5$; high: users with $vpd \geq 4.5$. The weekly distribution also shows the standard errors for the proportions.

4.3.3 Is There a "Universal" Rank Distribution?

We now move to address a fundamental question related to place transitions and how they are captured by manual and automatic check-in systems. Noulas et al. [30] recently introduced a model for human mobility that characterizes the transitions people make in urban areas. The intuition behind the model is that the probability of a person visiting, e.g., restaurant X does not depend on the distance to X but rather on the number of other restaurants that are closer to X, i.e. the *number of intervening opportunities* (= *rank*). Formally, the rank is defined as the number of places that are closer to the starting point of the transition than the destination of the transition. Furthermore, the model states that the distribution of ranks is universal across cities with different population densities, and it follows a power law for which an exponent of -0.88 was found in [30].

While Noulas et al. conducted their study with 4sq data from several cities across the globe, we study whether the power law also holds for the MDC data and the Swiss 4sq data set we have. In order to make the two data sets comparable, we look only into the transitions that happen within the French-speaking part of Switzerland since the MDC visits take place mainly there. As a rough approximation of this area, we consider only

places whose longitude is between 6° E and $7^\circ\,30'$ E (see Figure 4.2). We calculate the ranks of the transitions of a MDC user using only the user's own places, since if we included the MDC places of all users, we would sometimes have the same places incorrectly appearing multiple times. The rank distributions for MDC and 4sq are shown in Figure 4.7.

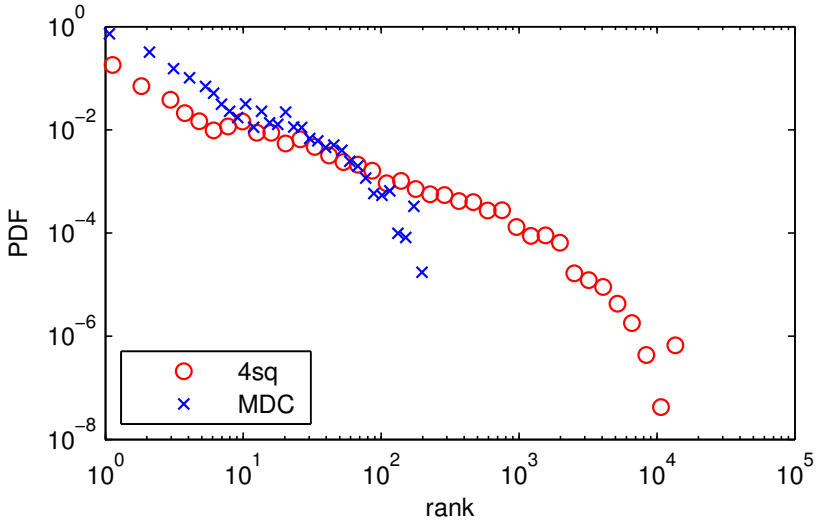

Figure 4.7: MDC and 4sq rank distributions which do not appear to follow the model proposed in [30].

The MDC distribution seems to have a steeper slope meaning that the MDC users are less likely to make transitions with a high rank. However, this is natural since a transition with rank 200 corresponds to a much longer travel distance for a MDC user than for a 4sq user. The reason for this is that the places of a single MDC user, while being "real" in the sense that they are the places that the user visits, actually are scattered much more sparsely than the places of all 4sq users.

We then redefine the rank of a MDC transition as the number of 4sq places that are closer to the starting point of the transition rather than the number of the user's own MDC places. The modified rank distribution is shown in Figure 4.8. Now the two distributions follow each other very closely. Up to rank 10^3, they are also in accordance with the universal rank distribution, estimated in [30] and shown with a dashed line, although with a slightly different exponent (-0.93) which has been estimated from the ranks below 10^3. However, from that on the distributions drop quickly to zero and do not follow a power law anymore. A natural explanation for this is that we consider only the transitions that happen within the

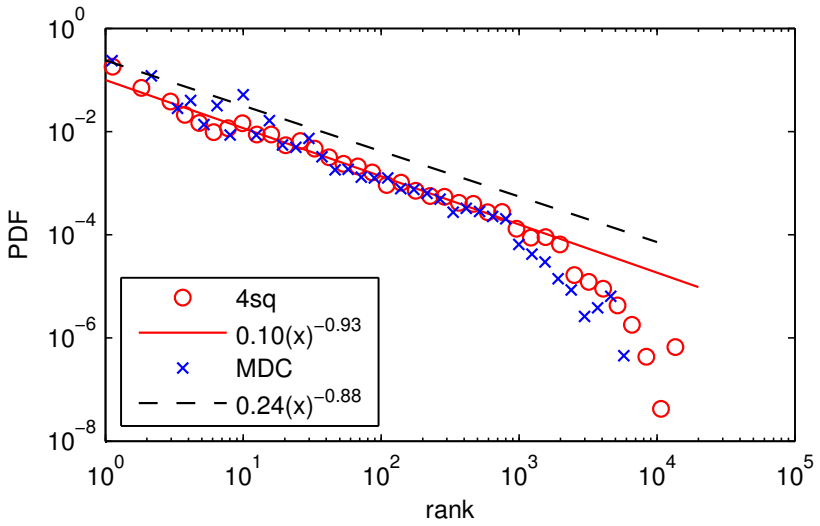

Figure 4.8: Rank distribution with the rank of a MDC transition defined as the number of 4sq places that are closer to the starting point of the transition than its end point. The dashed black line is the universal rank distribution given in [30].

western part of Switzerland. As a result, there cannot be many high rank transitions as there is not enough physical space to make them.

4.4 Matching MDC and Foursquare Places

So far we have investigated similarities and differences in temporal and transitional patterns for the MDC and 4sq data set. We now move to the problem of how to find correspondences across the data sets that could be useful for transfer learning purposes. More specifically, we want to study if it is possible to match the corresponding MDC and 4sq places.

4.4.1 Challenges in Finding the Matches

Let us take a look at MDC places and their nearby 4sq places. Figure 4.9 shows the distribution of distances from a MDC place to its closest 4sq place. We see that 50 % of the MDC places have at least one 4sq place within their radius of 100 meters, and 13 % have at least one 4sq place within 20 meters.

In general, the place matching problem is a difficult one. If there is only one 4sq place within 100 meters, it is trivial to match it to the MDC place although obviously we cannot be sure if they are really the same place.

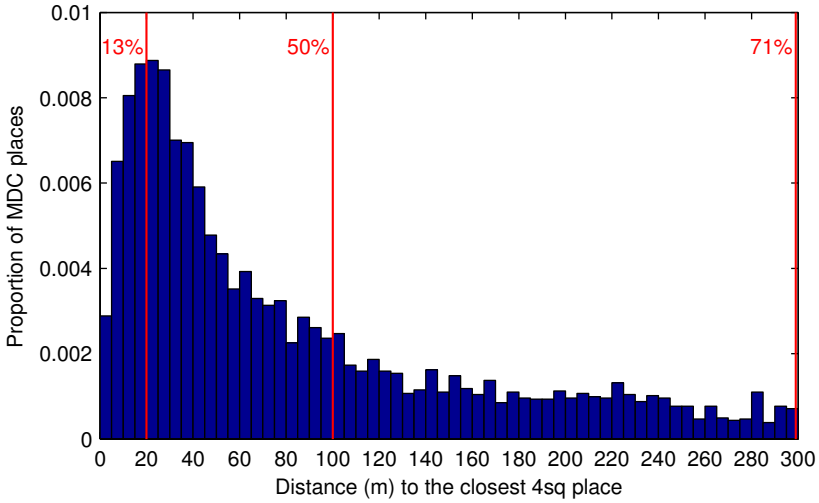

Figure 4.9: Distribution of distances from MDC places to their closest 4sq place. Vertical lines show the amount of probability mass on the left side.

However, in some cases there are more than one nearby 4sq place which we can see from Figure 4.10. Out of the 7281 MDC places, 33 % have at least two 4sq places within 100 meters; as an extreme case, one MDC place has 40 different 4sq places within 100 meters. In this kind of cases, it may be difficult to determine which one of the neighboring 4sq places actually corresponds to the MDC place.

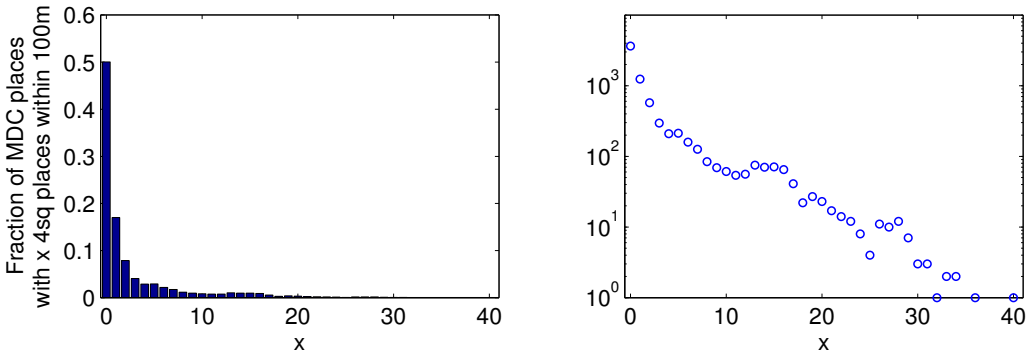

Figure 4.10: Left: probability distribution of x, where x is the number of 4sq places within a 100-meter radius from a MDC place. Right: the same distribution with the actual numbers of MDC places falling into each bin (instead of probabilities) and a logarithmic scaling for y-axis.

4.4.2 Nearest Neighbor Matching

Physical proximity is obviously a strong cue for matching places. We now consider a simple strategy where we always match the nearest 4sq place to each MDC place. We consider only the MDC places that have been labeled by the users (see Section 4.1) and that have at least one 4sq place within 100 meters. This corresponds to a total of 154 places.

To assess the accuracy of the method, we compare the MDC label to the category and title of the 4sq place and manually verify if they match. For instance, the MDC label *My workplace/school* and 4sq title/category pair *Batiment IN/College Engineering Building* would be considered a match. The mismatches, i.e., places whose function or meaning cannot be put in correspondence through manual inspection, are divided into two subcategories. The first subcategory *Fail1* refers to cases where the matched MDC and 4sq places are completely different, e.g., two nearby buildings. The second subcategory *Fail2* refers to cases where the MDC and 4sq places are inside the same building (e.g. two stores in the same mall) or the 4sq place is a subplace of the MDC place (e.g. a university cafeteria and the university). The motivation for this kind of distinction of mismatches is that while errors of the latter type (Fail2) are difficult to handle if we only use GPS data since GPS does not work well inside buildings, the errors of the first type (Fail1) may result from missing 4sq places. A 4sq place is missing if it has not been visited by the users in our data set during the data collection period of six months or if nobody has added it to 4sq, which is the case for a majority of the users' residences.

The results for the nearest neighbor place matching are shown in Table 4.3. We can see that a majority of the workplaces/schools (category C) are matched correctly. A typical example of a mismatched workplace/school is a MDC place that corresponds to the EPFL university while the nearest 4sq place corresponds to one of the EPFL cafeterias. Also *places for indoor sports* (category G) and *shops or shopping centers* (category I) have been matched accurately but for those categories we have less than 10 labeled MDC places. On the other hand, almost all of the *Homes* (category A and B) have been matched incorrectly, which is natural since 4sq users do not usually check in at their homes, and the likelihood of having MDC users who are at the same time 4sq users is very low given that the mobile platform used to record the MDC data (Nokia N95) is not very 4sq-friendly. If we exclude homes (categories A and B) and look at the places that are not isolated (i.e. have at least one 4sq place within 100 meters), we obtain a matching accuracy of 51 % for a total of 122 places. If we include all labeled MDC places, the matching accuracy drops to 19 %.

Table 4.3: Place matching results. Category and match type descriptions are shown in Table 4.4.

MDC Category	**Match**	Fail1	Fail2	Isolated	**Total**
A	2	21	0	63	86
B	0	9	0	37	46
C	42	11	22	29	104
D	7	8	3	4	22
E	0	2	1	6	9
F	4	4	0	17	25
G	4	1	0	9	14
H	0	5	0	6	11
I	5	3	0	9	17
J	0	0	0	5	5
Total	64	64	26	185	339

Table 4.4: Match type and category descriptions.

Type	Description
Match	Correct match.
Fail1	Matched MDC and 4sq places are completely different.
Fail2	MDC and 4sq places are different places but inside the same building.
Isolated	There are no 4sq places within 100 meters.

Category	Description
A	Home
B	Home of a friend, relative or colleague
C	My workplace/school
D	Location related to transportation (bus stop, metro stop, train station, parking lot, airport)
E	The workplace/school of a friend, relative or colleague
F	Place for outdoor sports (e.g. walking, hiking, skiing)
G	Place for indoor sports (e.g. gym)
H	Restaurant or bar
I	Shop or shopping center
J	Holiday resort or vacation spot

The relation between the distance to the closest 4sq place and the probability of correctly matching a place are illustrated in Figure 4.11. The figure shows that if the closest 4sq place is within 40 meters, it is probably a match, whereas after that mismatches are more probable. As a result, we can increase the precision of the matching by accepting only the 4sq places that are within 40 meters, whereas a threshold of 100 meters yields a higher recall. If we exclude homes (categories A and B) and look at the places that have at least one 4sq place within 40 meters, we obtain a matching accuracy of 67 % for a total of 60 places.

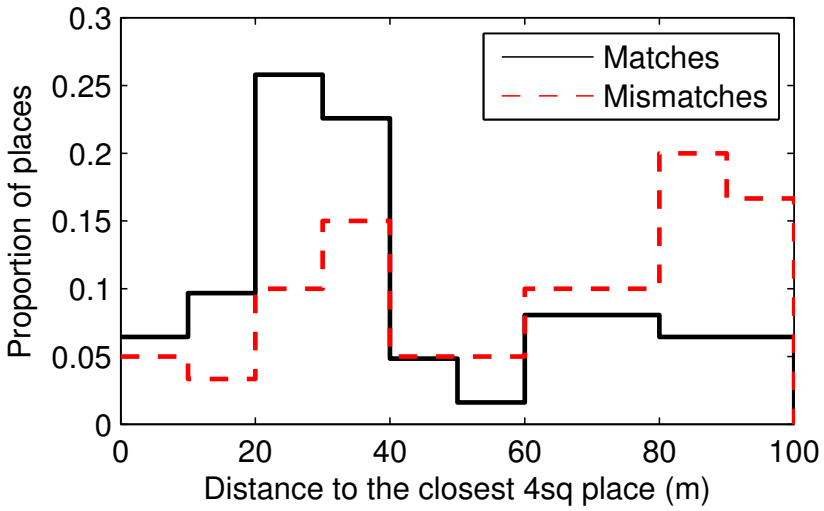

Figure 4.11: Distribution of distances to the closest 4sq place for the correctly matched MDC places and mismatched MDC places.

4.4.3 Improved Matching Methods

The previous subsection proposed a simple matching method based on the nearest 4sq place. To improve this approach, one could consider several nearest neighbors and make the choice between them based on other features than the distance. One alternative would be to look at the visit time distributions of the nearest neighbors and select the place whose distribution is closest to the visit time distribution of the MDC place, measured, e.g., by the Kullback–Leibler divergence [22]. However, this would work only for a minority of places since most of the 4sq places have been visited only a few times and thus it is not possible to estimate the visit time distribution for them. The distribution of the number of visits per 4sq place is shown in Figure 4.12 and it seems to follow a power law.

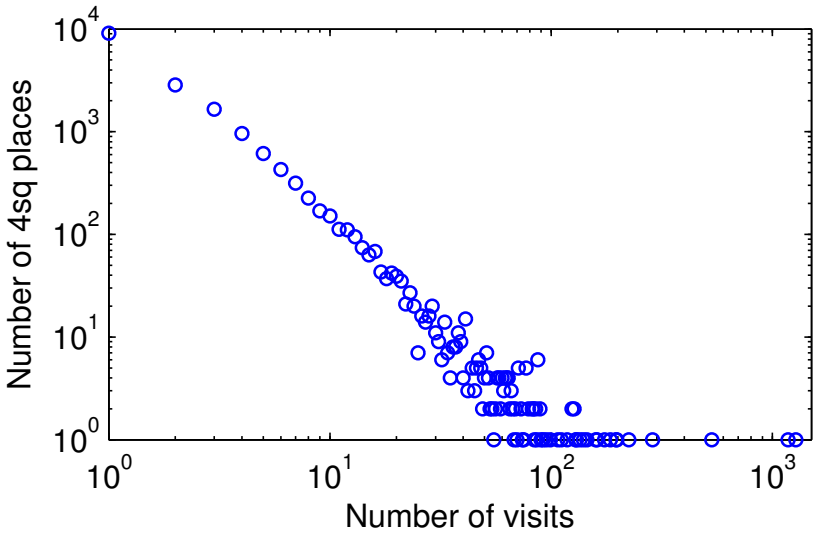

Figure 4.12: Distribution of the number of visits to 4sq places. On the y-axis, we have absolute counts of 4sq places.

Another additional feature that could be used to select one of the nearest neighbors is the semantic labels that were used to assess the performance of the nearest neighbor method in the previous subsection. As explained in Section 4.1, a few MDC places have been given one of the labels shown previously in Table 4.3. For the rest of the places, we could estimate the labels following one of the approaches proposed for the *MDC Semantic Place Prediction Task* (see proceedings of the MDC Workshop [24]). Then we could select the 4sq place whose place category (see Section 4.2) corresponds to the semantic label of the MDC place. A drawback of this method is that we would first need to manually define which MDC label corresponds to which 4sq category.

4.4.4 Implications for Transfer Learning

We have seen that finding the matches can be very difficult in some cases; 50 % of the MDC places do not have any 4sq place within their radius of 100 meters, in which case it is impossible to match the place. Furthermore, many places, e.g. in city centers, have between 2–40 nearby 4sq places which makes it hard to say which one of the nearby places is the correct match.

In consequence, the total matching accuracy using a simple nearest neighbor method is 19 %. Nevertheless, if we ignore the homes of the MDC users and consider only places which have at least one 4sq place

within 40 meters, the matching accuracy goes up to 67 %. This could be a sufficient performance in some applications if we were to consider, e.g., only workplaces/schools, which were matched relatively accurately. However, because of the low total matching accuracy and the fact that most 4sq places have only a few check-ins, we do not use place matching for transfer learning in this work. Instead we infer place categories and their check-in time distributions unsupervisedly from the 4sq data and do probabilistic matching between the MDC places and the inferred 4sq categories. This procedure is discussed in Chapter 6.

5

Next Place Prediction

In this chapter, we give a formal definition of the next place prediction task in a probabilistic framework. Then we present two state-of-the-art approaches, that won the *3rd* and the *1st* prize in the *MDC Next Place Prediction Task* [24]. Finally, we derive a novel approach for this task that is probabilistically rigorous and allows transfer learning using additional data sources that do not contain the end times of visits.

5.1 Problem Formulation

Our task is to find the most probable next place, i.e. the most probable value x of random variable X_{n+1}. We are given the current place $X_n = x'$ and the end time of the current visit $T_n^e = t$. Thus the task is formally given by

$$x_{\text{pred}} = \arg\max_x \{p(X_{n+1} = x \mid X_n = x', T_n^e = t)\}. \tag{5.1}$$

We shall denote the probability simply by

$$p(x_{n+1} \mid x_n, t_n^e). \tag{5.2}$$

The time of a visit consists of two parts, the hour and the weekday of the end time: $t^e = (h^e, d^e)$. Estimating the probability given in Equation 5.2 directly is problematic since we do not have enough data for obtaining a reliable estimate over all three random variables (x_n, h_n^e, d_n^e). Therefore, the approaches we present next make some simplifying assumptions for the probability.

5.2 State-of-the-Art Approaches

In this section, we present *Method I*, which is based on the approach by Gao et al. [15] and *Method II* which is based on the approach by Etter et al. [13]. These two teams won the *3rd* and the *1st* prizes in the MDC competition, respectively. Note that the probabilistic part of the method by the *2nd* team [39] is similar to Method II.

5.2.1 Method I

Gao et al. [15] start from Equation 5.2 and apply the Bayes' rule

$$p(x_{n+1} \mid x_n, t_n^e) = \frac{p(x_{n+1}, t_n^e \mid x_n)}{p(t_n^e \mid x_n)}. \tag{5.3}$$

Since we are only interested in finding the next place x_{n+1} that maximizes the probability, we can ignore the denominator and write

$$\begin{aligned} p(x_{n+1} \mid x_n, t_n^e) &\propto p(x_{n+1}, t_n^e \mid x_n) \\ &= p(x_{n+1} \mid x_n) p(t_n^e \mid x_{n+1}, x_n) \qquad (5.4) \\ &\approx p(x_{n+1} \mid x_n) p(t_n^e \mid x_{n+1}). \qquad (5.5) \end{aligned}$$

Note that Gao et al. make the assumption that the end time of the current visit does not depend on the current place given the next place, that is, $p(t_n^e \mid x_{n+1}, x_n) \approx p(t_n^e \mid x_{n+1})$. For example, if you always go to a hobby at a certain time of the week, you always need to leave the current place around the same time no matter where you are. However, this assumption does not take into account that if your current place is far from the hobby place, you have to reserve more time for reaching the next place on time.

The right-hand side of Equation 5.5 consists of two parts, a transition probability between two places and an end time distribution of the current place given the next place. Gao et al. propose to estimate the former using a hierarchical Bayesian n-gram model based on Pitman–Yor processes (HPY) [14, 37], but in this work, we use a *1st-order Markov model*. The advantage of the HPY model is that it incorporates smoothing and is thus also applicable to higher order n-gram models. However, in our problem setting, we are only given the current place and not the previous ones, and furthermore, the Markov model is more straightforward to learn which is why we choose to use it. We make the following improvement to the standard Markov model: in case the current place has not been visited previously, we use a *0-order Markov model*, i.e. the prior probabilities of

places, instead of a uniform distribution. This modification improves the predictions since there are lots of places that are visited only a few times and we know that when a person goes to such place, the next place will, nevertheless, be more likely a home than some random place the person has visited only rarely before.

The time distribution of Equation 5.5 can be split into an hour term and a weekday term if we assume that the visit hour and the visit day are independent

$$\begin{aligned} p(t_n^e \mid x_{n+1}) &= p(h_n^e, d_n^e \mid x_{n+1}) \\ &= p(h_n^e \mid d_n^e, x_{n+1}) p(d_n^e \mid x_{n+1}) \\ &\approx p(h_n^e \mid x_{n+1}) p(d_n^e \mid x_{n+1}). \end{aligned}$$

This approximation is not sound if a user, e.g., goes to a certain place always in the morning on one day and in the evening on another day of the week. However, the motivation for the approximation is that it reduces the number of parameters to be estimated by a factor of seven (number of weekdays). Nevertheless, even after this approximation there are places that have never been visited during a given hour, and therefore some smoothing is required [15]. Gao et al. propose to estimate distributions $p(h_n^e \mid x_{n+1})$ and $p(d_n^e \mid x_{n+1})$ using a Gaussian distribution. In addition, we compare this approach with multinomials and kernel density estimation that use Laplace smoothing.

The final form of the learned model is the following

$$p(x_{n+1} \mid x_n, t_n^e) \approx p(x_{n+1} \mid x_n) p(h_n^e \mid x_{n+1}) p(d_n^e \mid x_{n+1}). \tag{5.6}$$

Note that the right-hand side of the equation is unnormalized.

5.2.2 Method II

The winning method [13] in the MDC competition uses a blend of three methods: a Dynamical Bayesian Network (DBN), an Artificial Neural Network and a Gradient Boosted Decision Tree. We limit ourselves to the DBN model since the latter two methods are not probabilistic and thus they consider the prediction task as a multi-class classification task.

The proposed DBN model considers a spatial and a temporal component, similar to Method I (Equation 5.6), but it takes a mixture of these components

$$p(x_{n+1} \mid x_n, t_n^e) \approx \pi p(x_{n+1} \mid x_n) + (1 - \pi) p(x_{n+1} \mid h_n^e, w_n^e, \mathbb{I}_n^{\text{trust}}),$$

where w_n^e indicates whether the nth visit ends on a weekend or not and $\mathbb{I}_n^{\text{trust}}$ whether the transition between visits n and $n+1$ was trusted. We conduct our experiments only on the trusted transitions so the latter term can be ignored. Furthermore, h_n^e is a discretized end hour of visit n where a day is divided into k time periods. Etter et al. do not specify which value of k they use so we assume $k = 24$ which corresponds to one hour intervals.

Instead of directly estimating the temporal distribution, Etter et al. introduce variables h_{n+1}^s and w_{n+1}^s indicating the start time of the next visit in order to capture the randomness in the difference between the end time of the current visit and the start time of the next visit

$$\begin{aligned} p(x_{n+1} \mid h_n^e, w_n^e) &= \sum_{w_{n+1}^s} \sum_{h_{n+1}^s} p(x_{n+1}, h_{n+1}^s, w_{n+1}^s \mid h_n^e, w_n^e) \\ &= \sum_{w_{n+1}^s} \sum_{h_{n+1}^s} p(x_{n+1} \mid h_{n+1}^s, w_{n+1}^s, h_n^e, w_n^e) \\ &\quad\; p(h_{n+1}^s, w_{n+1}^s \mid h_n^e, w_n^e) \\ &\approx \sum_{w_{n+1}^s} \sum_{h_{n+1}^s} p(x_{n+1} \mid h_{n+1}^s, w_{n+1}^s) p(h_{n+1}^s, w_{n+1}^s \mid h_n^e, w_n^e) \end{aligned}$$

The resulting model is thus

$$\begin{aligned} p(x_{n+1} \mid x_n, t_n^e) \approx \pi p(x_{n+1} \mid x_n) + (1-\pi) \\ \sum_{w_{n+1}^s} \sum_{h_{n+1}^s} \left\{ p(x_{n+1} \mid h_{n+1}^s, w_{n+1}^s) p(h_{n+1}^s, w_{n+1}^s \mid h_n^e, w_n^e) \right\} \end{aligned} \quad (5.7)$$

Etter et al. present two ways of estimating the parameters of Equation 5.7. The first approach is to learn the three distributions by counting the frequencies of given realizations and then to optimize parameter π, while the second approach learns all parameters iteratively using the EM algorithm (see Sec. 3.3). We adopt the first approach and use a grid search to optimize π.

5.3 Our Approach

In this section, we introduce a novel approach to the next place prediction task (Equation 5.1) that combines the ideas of Method I and Method II. Like for Method I, we derive our approach starting from Equation 5.2 so that we can explicitly see which assumptions we make. We also want to incorporate the start time of the next visit instead of just using the end

time of the current visit as in Method II since this allows us to learn the time distributions from other data sources that do not have the end times of the visits.

Let us begin with Equation 5.4 for which we have not yet made any approximations

$$\begin{aligned} p(x_{n+1} \mid x_n, t_n^e) &\propto p(x_{n+1} \mid x_n) p(t_n^e \mid x_{n+1}, x_n) \\ &= p(x_{n+1} \mid x_n) \sum_{t_{n+1}^s} p(t_n^e, t_{n+1}^s \mid x_{n+1}, x_n) \\ &= p(x_{n+1} \mid x_n) \sum_{t_{n+1}^s} p(t_n^e \mid t_{n+1}^s, x_{n+1}, x_n) p(t_{n+1}^s \mid x_{n+1}, x_n). \end{aligned}$$

Now we have to make some simplifying assumptions for the two temporal components in order to reduce the number of parameters to be estimated. First we assume that

$$p(t_{n+1}^s \mid x_{n+1}, x_n) \approx p(t_{n+1}^s \mid x_{n+1}).$$

This is similar to what was done in Equation 5.5 but if we assume that the start time of a visit is usually more fixed than the end time, this assumption is more justified than the former one. For example, people may always go to work at certain time but then leave earlier or later depending on their next destination.

The randomness in the travel time between the current and the next place is captured by the other temporal component for which we make the following simplifying assumption

$$p(t_n^e \mid t_{n+1}^s, x_{n+1}, x_n) \approx p(\Delta t_n \mid \Delta x_n), \tag{5.8}$$

where Δt_n is the time difference between t_n^e and t_{n+1}^s in hours, and Δx_n is the distance between places x_n and x_{n+1} in kilometers. That is to say, the travel time between two consecutive visits does not depend on the time of the day or the actual place where a user is or is going to but only on the distance between the places. In some cases the assumption is intuitively unsound as traveling a certain distance is usually slower in a city center than in a less densely populated area. Nevertheless, it is sufficient that the assumption holds within a one hour window, which is the time resolution of the model.

The final form of the proposed model for next place prediction is given by

$$\begin{aligned} &p(x_{n+1} \mid x_n, t_n^e) \approx \\ &\quad p(x_{n+1} \mid x_n) \sum_{t_{n+1}^s} \left\{ p(\Delta t_n \mid \Delta x_n) p(h_{n+1}^s \mid x_{n+1}) p(d_{n+1}^s \mid x_{n+1}) \right\}. \end{aligned} \tag{5.9}$$

5.4 Experimental Results

In Section 5.4.1, we optimize parameter π of Method II and learn the travel time distribution $p(\Delta t_n \mid \Delta x_n)$ of our approach, and in Section 5.4.2, we present the next place prediction results.

5.4.1 Parameter Optimization

We learn the mixing coefficient parameter π (see Equation 5.7) using a grid search. Only the first 80 % of each user's visits (*Set A*) can be used to learn the parameter. This is because Set A is later used as the training data for the next place prediction, while the remaining 20 % (*Set T*) are reserved for calculating the final test accuracies (= proportion of correctly predicted next places). Hence, we split Set A into *Set B*, which contains the first 80 % of the training samples and is the new training set, and *Set C*, which contains the last 20 % of the training samples and is the new test set. The data set division is illustrated in Figure 5.1.

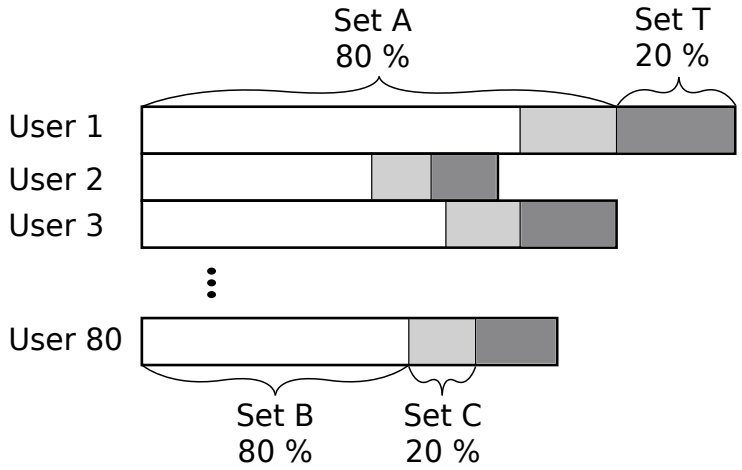

Figure 5.1: The check-in sequences of the MDC users are divided into different sets used for training, validation, and testing.

The individual distributions of Equation 5.7 are learnt from Set B while varying the values of π from 0 to 1, and validation accuracies are calculated on Set C. Note that cross-validation is not appropriate in our case since the samples are not independent but we always predict future transitions. The results are shown in Figure 5.2. Based on these results, we set $\pi = 0.14$.

Next let us study the justifiability of the assumption made in Equation 5.8 by looking at the actual travel times and travel distances found

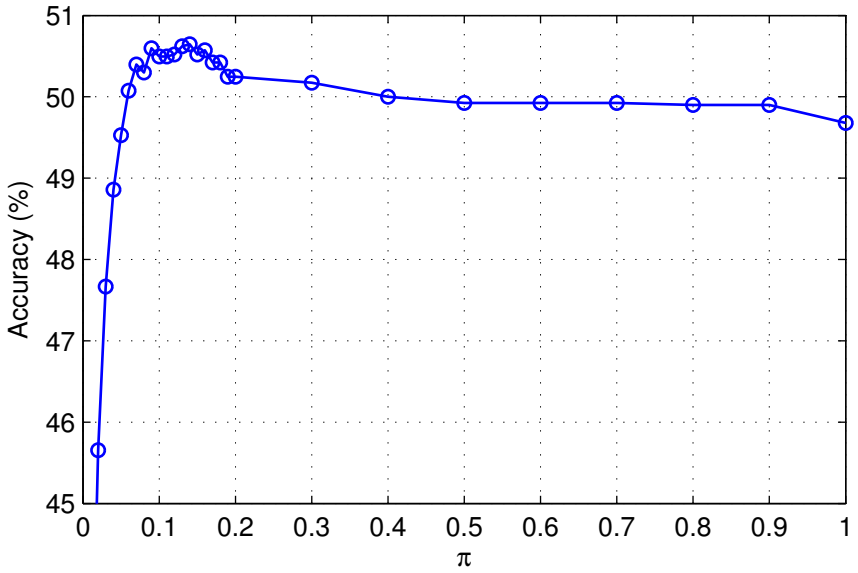

Figure 5.2: Next place prediction accuracy as a function of the mixing coefficient π (see Equation 5.7).

in the MDC data. The two variables are plotted against each other in Figure 5.3 using Set A. The figure shows that the two variables are pos-

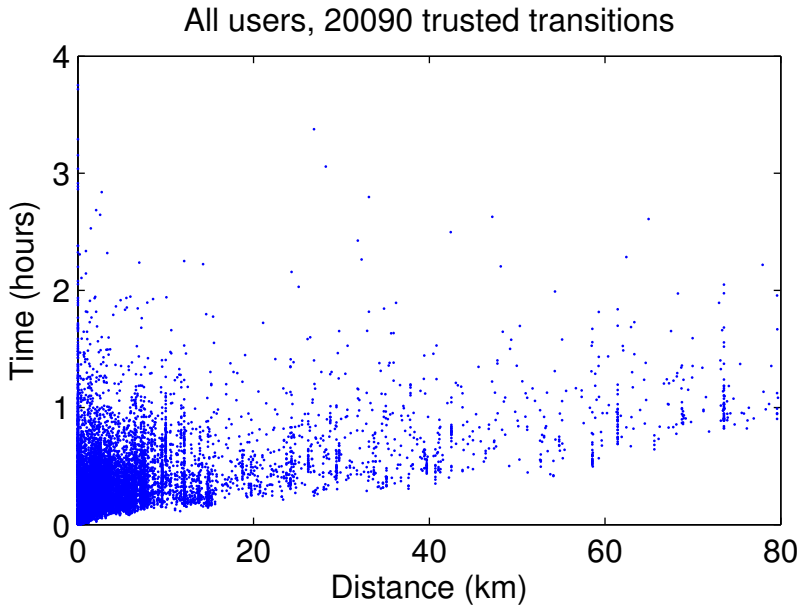

Figure 5.3: Travel distances and the corresponding travel times between two consecutive visits for all users.

itively correlated since the travel time increases when the distance increases. Pearson's linear correlation coefficient is $\rho = 0.38$ and the p-value for the hypothesis of no correlation is 0.00. Even though the data points are rather dispersed in the vertical direction, they are still quite strongly determined within our model's time resolution of one hour given only the travel distance. Therefore, we consider this model reasonably accurate for our purposes.

Figure 5.4 shows the actual travel time model. We have discretized the travel distance into 5 kilometer bins, and we consider the travel time only up to four hours since above that the probability is effectively zero. The model shows how the travel time depends on the travel distance. If the transition is less than 25 kilometers, the travel time is shorter than one hour with over 90% probability.

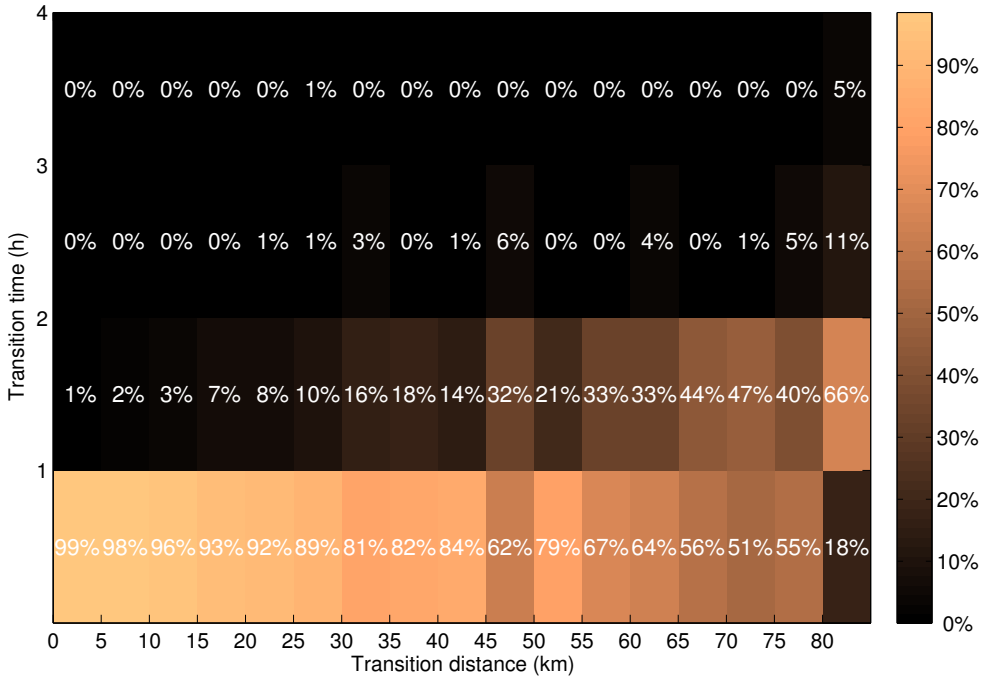

Figure 5.4: Travel time model $p(\Delta t_n \mid \Delta x_n)$ estimated from Set A.

5.4.2 Prediction

We compare the performance of the different methods using Set T as the test data and Set A as the training data (see Figure 5.1). Note that the test data has not been used to learn any of the model parameters. The comparison is done between Method I, Method II, and our approach. Further-

more, for Method I, we test three different strategies for learning the time distributions: the Gaussian distribution, as originally proposed by Gao et al., multinomial model, and kernel density estimation. These strategies are presented in detail in Section 3.4. For Method II and for our approach, we use only the multinomial model. The results are shown in Figure 5.5 and the abbreviations are explained in Table 5.1.

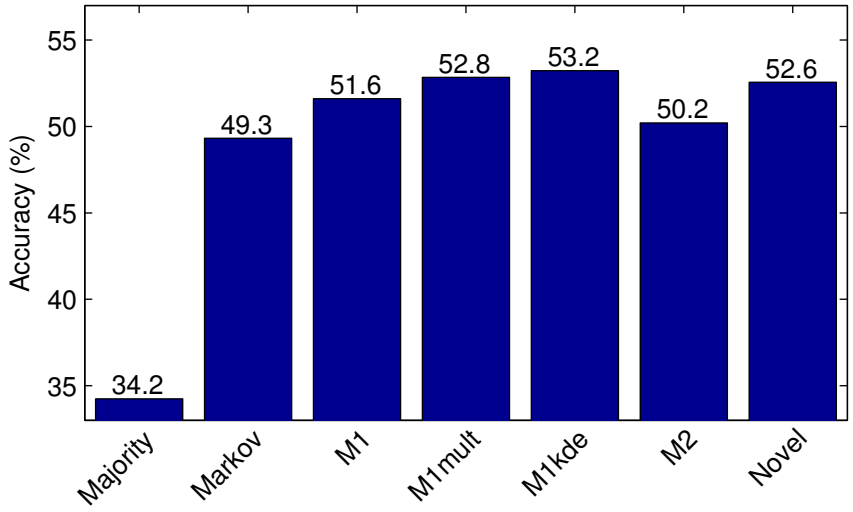

Figure 5.5: Performance of all the studied methods.

Table 5.1: Method descriptions.

Abbreviation	Description
Majority	Majority vote which predicts the most frequent place.
Markov	1st-order Markov model.
M1	Method I using Gaussian distributions (see Section 5.2.1).
M1mult	Method I using multinomial distributions.
M1kde	Method I using kernel density estimation.
M2	Method II (see Section 5.2.2).
Novel	Our approach (see Section 5.3).

The results show that there are clearly some spatial visit patterns in the data, i.e. the next location of a person largely depends on the current location, since the Markov model outperforms the Majority model. We can also see that time plays a role in determining the next place, as the remaining methods, which consider also the temporal context, perform better than the Markov model.

The prediction accuracy of the original M1 method can be improved by using multinomials or kernel density estimation for the check-in time distributions. This means that the normality assumption for check-in times does not hold. Somewhat surprisingly, the performance of M2, the probabilistic part of the winning method, is lower than the performance of M1. This suggests that the secret behind the success of the winning method is that it combines probabilistic and discriminative classifiers and/or that it implements minor improvements, such as the detection of the change of residential location [13].

Our approach performs slightly worse than M2mult, which is its closest counterpart. The difference between the two methods is that our approach sums over the different possible travel times from the current place to the next place. We may conclude that either the assumption made in Equation 5.8 is too strong or that we should find a way to estimate the travel time distribution more accurately. Nevertheless, our approach outperforms both of the two originally proposed probabilistic methods M1 and M2 and its advantage is that we can use it to transfer knowledge from 4sq since it does not require the the information about the end times of visits. Note that the performance of M1 could potentially be improved by using the HPY model instead of the Markov model as discussed in Section 5.2.1. Nevertheless, a similar improvement could be expected in the case of our approach if we incorporated the HPY model.

We conclude by looking at how much the prediction accuracy varies over users. Figure 5.6 shows the accuracy distribution for our approach. The most predictable user has an accuracy of 85.9 % (92 test samples). On the other hand, there are two users whose accuracy is zero, which is partly explained by their lack of data (8 and 1 test samples). Overall, the accuracies are reasonably high given that the average number of different places per user is 51, which implies an accuracy of 2 % for random prediction. Furthermore, it is natural that we do not observe accuracies close to 100 % since there are other factors affecting human mobility than just the current place and time.

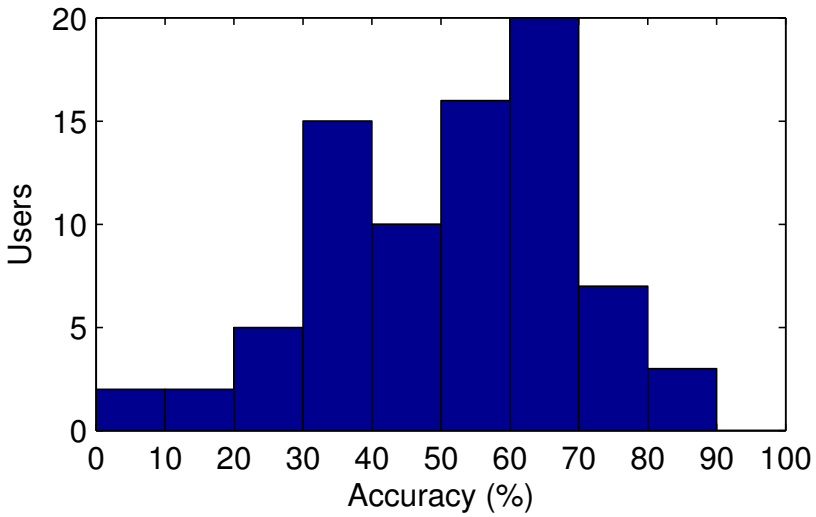

Figure 5.6: Next place prediction accuracies for different users.

6

Transfer Learning of Time Distributions

A crucial part in modeling human mobility patterns is to model the temporal characteristics of a place. Some places, such as schools and offices, are typically visited on weekdays before noon, whereas others, such as bars and nightclubs, are visited more towards the end of the week in the evening. In this chapter, we study these temporal differences of places belonging to different categories.

In 4sq, the places have been labeled under predefined categories, such as *restaurant*, *movie theater*, *office building*, etc. The check-in time distributions of these categories are analyzed in Section 6.1. In Section 6.2, we introduce a mixture model for finding temporally separate place categories in an unsupervised manner, and in Section 6.3, we show how the mixture model together with a posterior predictive distribution can be used to infer the time distribution of a place from two data sets. Finally, we present experimental results regarding transfer learning of the time distributions in Section 6.4.

6.1 Foursquare Place Categories

The 4sq places are divided into the following nine main categories:

1. Arts & Entertainment
2. College & University
3. Food
4. Nightlife Spot

5. Outdoors & Recreation
6. Professional & Other Places
7. Residence
8. Shop & Service
9. Travel & Transport

These main categories are further divided into subcategories and sometimes also into subsubcategories[1], e.g., *Outdoors & Recreation* → *Athletic & Sport* → *Hockey Field*.

In order to study the temporal characteristics of the 4sq categories, we collect the check-ins of all places belonging to a category and plot the daily and weekly time distributions of the check-ins. The daily distributions are shown in Figure 6.1. The findings support our intuitive notions about these categories: *nightlife spots* are visited mainly between 18^h and 24^h, whereas the peak hour for *Colleges & Universities* is at 9^h; *Food* places are most active at noon when people have lunch and around 19^h when people have dinner; places related to *Travel & Transport* are active throughout the day but quiet down for the night. Furthermore, the distributions give us insights into the Swiss culture. For example, we can notice a sudden drop in the check-in activity of *Shops & Services* after 19^h when most of the shops close in Switzerland.

We have also plotted the weekly check-in time distributions in Figure 6.2. These plots reveal some categories that are active mainly on weekends (1, 5), on Saturdays but not Sundays (4, 8), on weekdays (2, 6), and categories whose activity levels do not vary much over the week (3, 7, 9).

The time distributions vary across categories but there is also a lot of variation within categories since some of the categories enclose a diverse set of subcategories. For instance, category *Travel & Transport* contains mainly places where one waits for a transportation mean (airports, train stations, bus stops) but it also contains subcategory *Hotel*. The daily check-in time distributions of hotels and other places belonging to category 9 are shown in Figure 6.3. The distribution for hotels is completely missing the activity peek around noon.

[1]The full list of categories can be found at `https://developer.foursquare.com/docs/venues/categories`

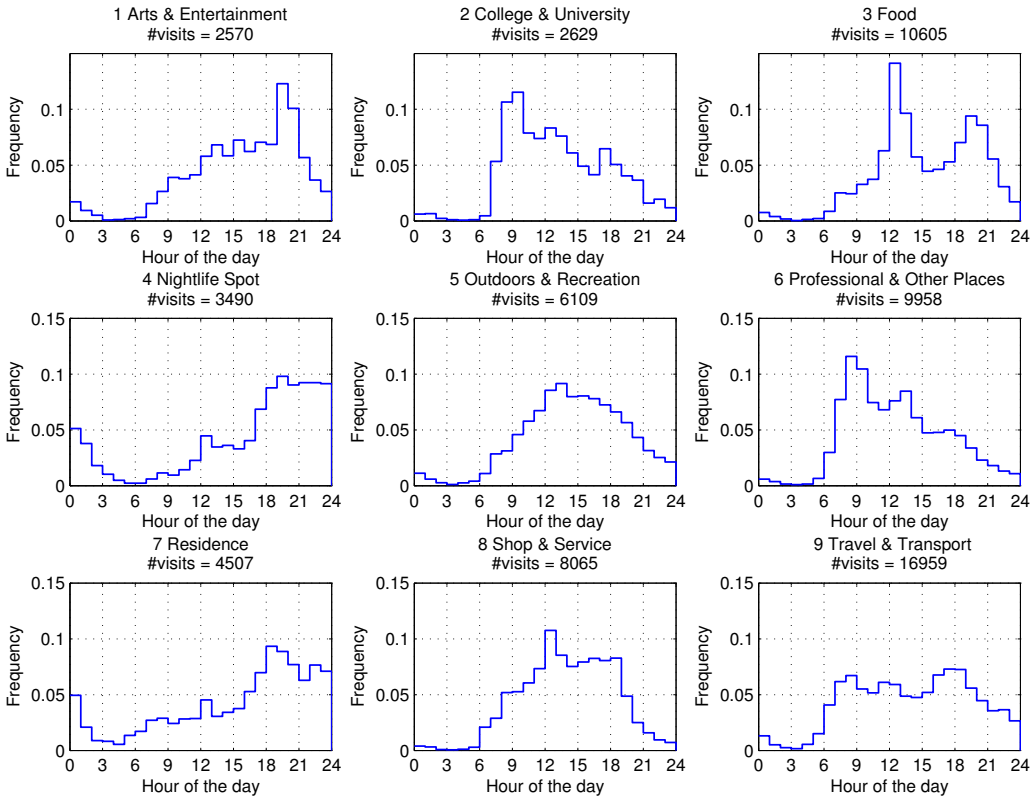

Figure 6.1: Daily distributions of check-ins in different 4sq categories.

6.2 Unsupervised Category Inference

In the previous section, we saw that different 4sq categories exhibit significantly different check-in time distributions. However, the 4sq category alone does not sufficiently determine the time distribution of a place since there is lots of variation within a category. Therefore, we would like to find the underlying place categories so that within a category all places would follow the same time distribution.

This problem can be seen as a clustering problem where each cluster is represented by a time distribution model and the deviations from the model within a cluster are to be minimized. What distinguishes this from a traditional clustering problem is that normally the samples to be clustered are individual data points, but in our case they are places containing an indefinite number of check-in times.

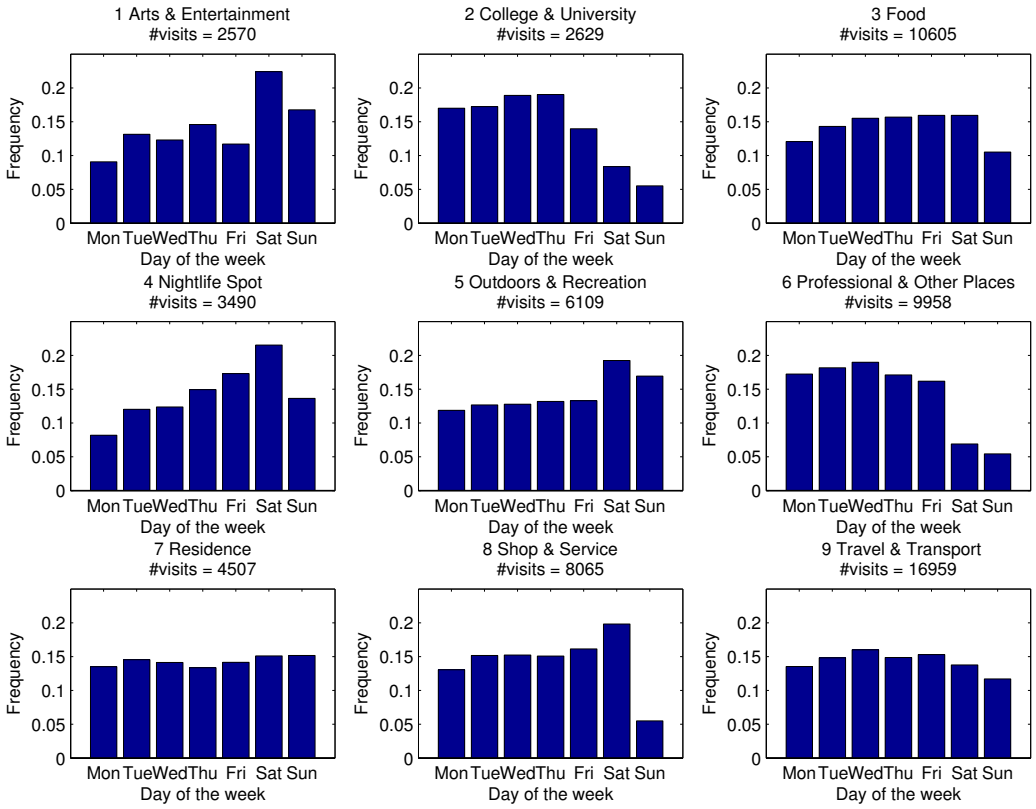

Figure 6.2: Weekly distributions of check-ins in different 4sq categories.

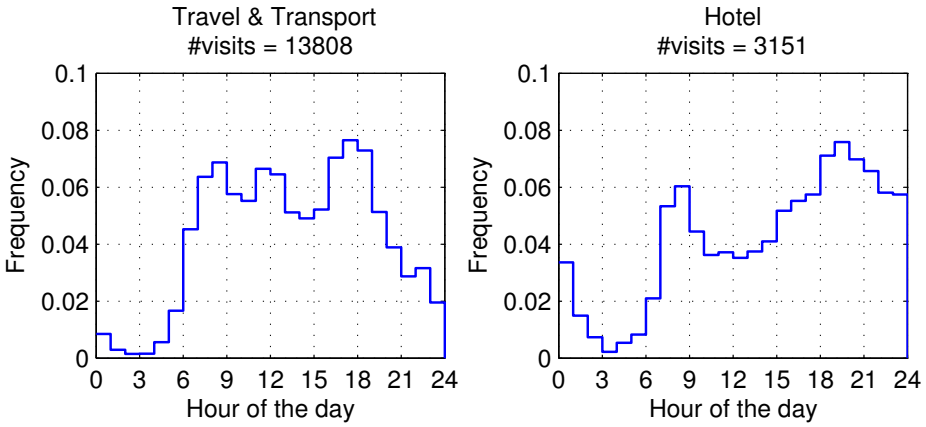

Figure 6.3: Daily check-in time distributions of all places in the category called *Travel & Transport* except for hotels (left) and all places in the subcategory called *Hotel* (right).

6.2.1 Mixture of Multinomials

The approach we take to solve the place clustering problem is to learn a mixture model for the data so that each component of the mixture corresponds to a cluster. We introduce latent variables that indicate which component has generated the check-ins of a place. The check-in times are considered discrete hour-weekday pairs, and thus we use a multinomial distribution (see Section 3.4.1) for each component. That is, we model the data with a *mixture of multinomials*.

A check-in time t consists of a discrete hour and weekday: $t = (h, d) \in (\{0, 1, \ldots, 23\} \times \{1, 2, \ldots, 7\})$. Furthermore, we assume that the hour and weekday are independent, i.e., $p(h, d) = p(h)p(d)$. We denote a vector of all check-in times from a single place by $\mathbf{t}$. Alternatively, it can be represented by check-in counts $(\mathcal{H}, \mathcal{D})$, where $\mathcal{H}_i$ is the number of check-ins occurring at hour i and $\mathcal{D}_j$ is the number of check-ins occurring at weekday j. From Equation 3.5, we obtain

$$p(\mathbf{t} \mid \boldsymbol{\theta}, \boldsymbol{\varphi}) = \sum_{c=1}^{C} \pi_c \frac{N!}{\prod_{i=0}^{23} \mathcal{H}_i!} \left(\prod_{i=0}^{23} \theta_{ci}^{\mathcal{H}_i} \right) \frac{N!}{\prod_{j=1}^{7} \mathcal{D}_j!} \left(\prod_{j=1}^{7} \varphi_{cj}^{\mathcal{D}_j} \right), \tag{6.1}$$

where C is the number of components, π_c is the mixing coefficient for component c, N is the number of check-ins to the place, and $\boldsymbol{\theta}_c$ and $\boldsymbol{\varphi}_c$ are the parameters for the daily and weekly distribution of component c, respectively.

6.2.2 Parameter Estimation via EM Algorithm

Let us denote the check-in time vectors of all places by $\mathbf{T}$ which is the observed data. The log likelihood function takes the form

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\varphi} \mid \mathbf{T}) &= \log \prod_{l=1}^{L} p(\mathbf{t}_l \mid \boldsymbol{\theta}, \boldsymbol{\varphi}) \\ &= \sum_{l=1}^{L} \log \sum_{c=1}^{C} \pi_c \frac{N_l!}{\prod_{i=0}^{23} \mathcal{H}_{li}!} \left(\prod_{i=0}^{23} \theta_{ci}^{\mathcal{H}_{li}} \right) \frac{N_l!}{\prod_{j=1}^{7} \mathcal{D}_{lj}!} \left(\prod_{j=1}^{7} \varphi_{cj}^{\mathcal{D}_{lj}} \right), \end{aligned}$$

where L is the number of places, N_l the number of check-ins to place l, and $\mathcal{H}_{li}$ and $\mathcal{D}_{lj}$ are the numbers of check-ins to place l occurring at hour i and weekday j, respectively. Denoting $A_l = \frac{N_l! N_l!}{\prod_{i=0}^{23} \mathcal{H}_{li}! \prod_{j=1}^{7} \mathcal{D}_{lj}!}$, we obtain a

simpler form

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\varphi} \mid \mathbf{T}) = \sum_{l=1}^{L} \left[\log(A_l) + \log \sum_{c=1}^{C} \pi_c \left(\prod_{i=0}^{23} \theta_{ci}^{\mathcal{H}_{li}} \right) \left(\prod_{j=1}^{7} \varphi_{cj}^{\mathcal{D}_{lj}} \right) \right]. \tag{6.2}$$

Setting the partial derivatives of the likelihood function to zero yields a system of equations which cannot be solved in a closed form. Therefore, we need to use another approach in order to find the maximum likelihood parameter estimates, and in this work, we choose to use the EM algorithm (see Section 3.3) which is an iterative approach to find a maximum likelihood solution. We start by initializing all parameter values $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$ randomly.

In the **E step**, we calculate the posterior distribution of the latent variable z_{lc} which is a $0/1$ hidden variable indicating whether the check-in times of place l were generated by component c

$$p(z_{lc} = 1 \mid \mathbf{t}_l, \boldsymbol{\theta}, \boldsymbol{\varphi}) = \frac{\pi_c \left(\prod_{i=0}^{23} \theta_{ci}^{\mathcal{H}_{li}} \right) \left(\prod_{j=1}^{7} \varphi_{cj}^{\mathcal{D}_{lj}} \right)}{\sum_{c'=1}^{C} \pi_{c'} \left(\prod_{i=0}^{23} \theta_{c'i}^{\mathcal{H}_{li}} \right) \left(\prod_{j=1}^{7} \varphi_{c'j}^{\mathcal{D}_{lj}} \right)} \equiv \gamma_{lc}. \tag{6.3}$$

In the **M step**, we re-estimate multinomial distribution parameters for each component. The update equations for the parameters are given by

$$\theta_{ci}^{\text{new}} = \frac{0.1 + \sum_{l=1}^{L} \gamma_{lc} \mathcal{H}_{li}}{2.4 + L_c}, \quad i = 1, 2, \ldots, 24 \tag{6.4}$$

$$\varphi_{cj}^{\text{new}} = \frac{0.1 + \sum_{l=1}^{L} \gamma_{lc} \mathcal{D}_{lj}}{0.7 + L_c}, \quad j = 1, 2, \ldots, 7 \tag{6.5}$$

$$\pi_c = \frac{L_c}{L}, \tag{6.6}$$

where $L_c = \sum_{l=1}^{L} \gamma_{lc}$ and constants 0.1, 2.4, and 0.7 are due to additive smoothing.

The E and M step are repeated for a fixed number of iterations or until the change in the log likelihood is less than a given threshold. The derivations of the E and M step with discussion about numerical challenges related to the above equations are presented in Appendix A.

6.3 Inferring Time Distributions Across Data Sets

Let us look how to estimate the time distribution of a place with only a few check-ins using a mixture of multinomials learned from another check-in

data set. We assume that the underlying place categories are the same in the two data sets and introduce latent variable z which indicates the component (= category) that has generated the previous check-in times $\mathbf{t}$. The *Posterior Predictive Distribution (PPD)* of the next check-in time $\tilde{t}$ takes the form

$$\begin{aligned}
p(\tilde{t} \mid \mathbf{t}, \boldsymbol{\theta}, \boldsymbol{\varphi}) &= \sum_{z} p(\tilde{t}, z \mid \mathbf{t}, \boldsymbol{\theta}, \boldsymbol{\varphi}) \\
&= \sum_{c=1}^{C} p(\tilde{t} \mid z = c, \mathbf{t}, \boldsymbol{\theta}, \boldsymbol{\varphi})\, p(z = c \mid \mathbf{t}, \boldsymbol{\theta}, \boldsymbol{\varphi}) \\
&\approx \sum_{c=1}^{C} p(\tilde{t} \mid z = c, \boldsymbol{\theta}, \boldsymbol{\varphi})\, p(z = c \mid \mathbf{t}, \boldsymbol{\theta}, \boldsymbol{\varphi}) \\
&\propto \sum_{c=1}^{C} p(\tilde{t} \mid z = c, \boldsymbol{\theta}, \boldsymbol{\varphi})\, p(z = c \mid \boldsymbol{\theta}, \boldsymbol{\varphi})\, p(\mathbf{t} \mid z = c, \boldsymbol{\theta}, \boldsymbol{\varphi}) \\
&\approx \sum_{c=1}^{C} p(\tilde{h} \mid \boldsymbol{\theta}_c) p(\tilde{d} \mid \boldsymbol{\varphi}_c) \pi_c \frac{N!}{\prod_{j=0}^{23} \mathcal{H}_j!} \left(\prod_{i=0}^{23} \theta_{ci}^{\mathcal{H}_i} \right) \frac{N!}{\prod_{j=1}^{7} \mathcal{D}_j!} \left(\prod_{j=1}^{7} \varphi_{cj}^{\mathcal{D}_j} \right) \\
&\propto \sum_{c=1}^{C} p(\tilde{h} \mid \boldsymbol{\theta}_c) p(\tilde{d} \mid \boldsymbol{\varphi}_c) \pi_c \left(\prod_{i=0}^{23} \theta_{ci}^{\mathcal{H}_i} \right) \left(\prod_{j=1}^{7} \varphi_{cj}^{\mathcal{D}_j} \right), \qquad (6.7)
\end{aligned}$$

where $p(\tilde{h} \mid \boldsymbol{\theta}_c) = \prod_{i=0}^{23} \theta_{ci}^{\tilde{\mathcal{H}}_i}$ when we transform the next check-in hour $\tilde{h}$ into count representation $\tilde{\mathcal{H}}$, and similarly for $p(\tilde{d} \mid \boldsymbol{\varphi}_c)$. Note that we have approximated $p(z = c \mid \boldsymbol{\theta}, \boldsymbol{\varphi}) = p(z = c) \approx \pi_c$. This holds only if the relative proportions of different categories are the same in MDC and 4sq, since π_c has been estimated based on 4sq data. This is not the case, e.g., for a category containing homes, but we assume that the proportions are similar enough. Comparing Equations 6.7 and 6.1, the main difference is that the mixing coefficients π_c are now accompanied by expression $(\prod_{i=0}^{23} \theta_{ci}^{\mathcal{H}_i})(\prod_{j=1}^{7} \varphi_{cj}^{\mathcal{D}_j})$ which measures how well the previous observations fit to component c. To give an idea what the PPD does in practice, we show two examples of multinomial distributions estimated directly from the previous MDC visits and the corresponding PPDs that use the 4sq mixture model in Figure 6.4. We can see that the PPD smooths the multinomial distribution, but on the other hand, the effect of Laplace smoothing disappears.

In our problem setting, we learn parameters $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$ from 4sq. In addition to the PPD, we also want to include the *MDC only* model $(\boldsymbol{\theta}^{\text{MDC}}, \boldsymbol{\varphi}^{\text{MDC}})$ which encodes a single multinomial estimated directly from the previous

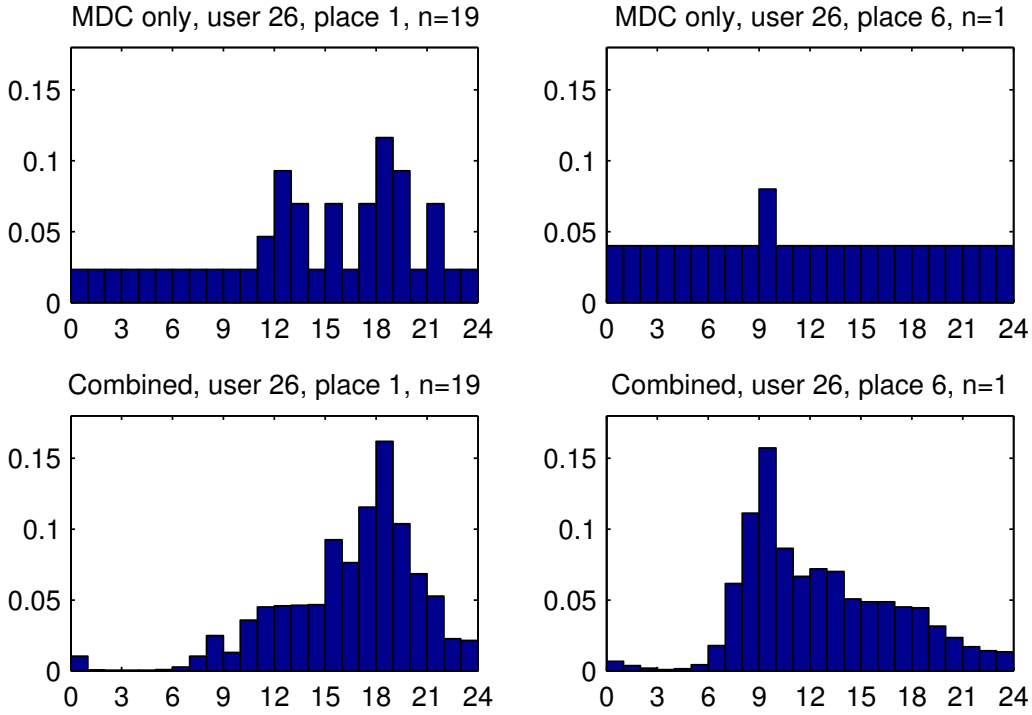

Figure 6.4: Multinomial check-in hour distributions estimated directly from the previous MDC visits (above) and the corresponding posterior predictive distributions that use the 4sq mixture model (below).

MDC visits. The reason is that the MDC and 4sq data sets do not follow exactly the same distribution and if a MDC place has been visited many times in the past, a direct estimate will be accurate. The two models are combined as follows

$$p(\tilde{t} \mid \mathbf{t}, \boldsymbol{\theta}^{\text{MDC}}, \boldsymbol{\varphi}^{\text{MDC}}, \boldsymbol{\theta}^{\text{4sq}}, \boldsymbol{\varphi}^{\text{4sq}}) = \\ \alpha\, p(\tilde{t} \mid \boldsymbol{\theta}^{\text{MDC}}, \boldsymbol{\varphi}^{MDC}) + (1-\alpha)\, p(\tilde{t} \mid \mathbf{t}, \boldsymbol{\theta}^{\text{4sq}}, \boldsymbol{\varphi}^{\text{4sq}}), \tag{6.8}$$

where α is a transfer coefficient determining how much weight is given for the MDC only model and how much for the PPD model learned from both MDC and 4sq data. We either keep α fixed or let it depend on the number of previous visits (N_p) to the MDC place.

A fixed α is straightforward to optimize: we use a grid search to find α that maximizes validation accuracy. That is, we calculate the prediction accuracy for values $\alpha = \frac{0}{n-1}, \frac{1}{n-1}, \ldots, \frac{n-1}{n-1}$ and select the best α. In case of the varying α model, the optimization task is considerably more challenging since we need to find the best value of α for each N_p. If we were to consider m different values of N_p and n values of α, the resulting number

of iterations in a grid search would be n^m which is an infeasible complexity even for small values of n and m since calculating the predictions for all 80 users once takes about one minute. One possibility to make the estimation of $\alpha(N_p)$ easier is to find an appropriate curve with only a few parameters to fit. However, what we can assume about the functional form of $\alpha(N_p)$ is that it is a monotonically increasing function since the more data we have, the more accurate will our MDC only model be.

We choose to use a piecewise linear function to estimate $\alpha(N_p)$ and set the nodes of the function to $N_p = 0, 5, 10, \ldots, 5(m-1)$. The parameter space is still too vast so we make an assumption that the optimal α value of a node depends only on the previous nodes. This allows us to learn the nodes incrementally reducing the complexity from $\mathcal{O}(n^m)$ to $\mathcal{O}(nm)$. In practice, we need even less iterations since we only have to consider those α values that are equal or higher than the previous node. Furthermore, the maximum value of a node is 1, so once a node gets value 1, we can terminate the search since the rest of the nodes will also be 1 due to the monotonicity. This greedy heuristic is fast but it does not guarantee a global optimum. The approach is presented in Algorithm 6.3.1 and also illustrated later in the results section in Figure 6.7.

Algorithm 6.3.1 Greedy heuristic for optimizing piecewise linear $\alpha(N_p)$

Input: n is the number of α values and m is the number of nodes in the piecewise linear function

Output: Array $alphas$ contains the α values at the optimized nodes

```
1  alphas(i) = 1,    i = 1, 2, ... m
2  previous_alpha_idx = 1
3  for i from 1 to m do
4      best_accuracy = −1
5      best_idx = −1
6      for j from previous_alpha_idx to n do
7          alphas(i) = (j−1)/(n−1)
8          accuracy = DoPredictions(alphas)
9          if accuracy > best_accuracy then
10             best_accuracy = accuracy
11             best_idx = j
12     alphas(i) = (best_idx−1)/(n−1)
13     previous_alpha_idx = best_idx
14     if previous_alpha_idx == n then
15         break
```

6.4 Experimental Results

In Sections 6.4.1 and 6.4.2, we optimize the parameters of the models needed for transfer learning and in Section 6.4.3, we quantify the effect of transfer learning on the next place prediction problem.

6.4.1 Number of Underlying Categories

We want to learn a mixture model that has a high predictive performance, i.e., it describes well not only the data it was trained with but also unseen test data. The more components (= categories) we give our mixture model, the more complex distribution it can capture. However, with only a limited number of training samples, we face the problem of overfitting if the number of components is too high.

There are a number of approaches to find the optimal number of components in a mixture model, such as the *Akaike's information criterion* and *Bayesian model selection* [1]. In this work, we use *k-fold cross-validation*. In this approach, the data is divided into k folds and, one at a time, each of these folds is held-out as a test set while the remaining folds are used for training. Cross-validation likelihood is calculated by taking the average of k test likelihoods. This allows us to make use of all data for assessing the predictive power of our mixture model.

We learn the mixture model based on the 4sq data set. First, we ignore all 4sq places which have less than five check-ins, which leaves us with 2888 places out of the total of 17631 places. Then we divide the places into ten folds and run the cross-validation procedure. The average log likelihood of the training folds and test folds are shown in Figure 6.5.

We want to show that these components learned in an unsupervised manner are in fact better than the original 4sq categories. Therefore, we have calculated the test likelihoods using the nine main categories and all 252 separate 4sq categories. The test likelihoods are calculated in two ways: based on only the model of each test place's own category and based on a mixture model which sets the values of the latent variables according to the 4sq category labels in the training phase. These results are also given in Figure 6.5.

Based on the results, we can first of all say that an unsupervised approach yields better categories with respect to the similarity of the time distributions of places within the same category than if we used the predefined 4sq categories. Secondly, we can determine the optimal number of components in the mixture model by looking at the validation likelihood

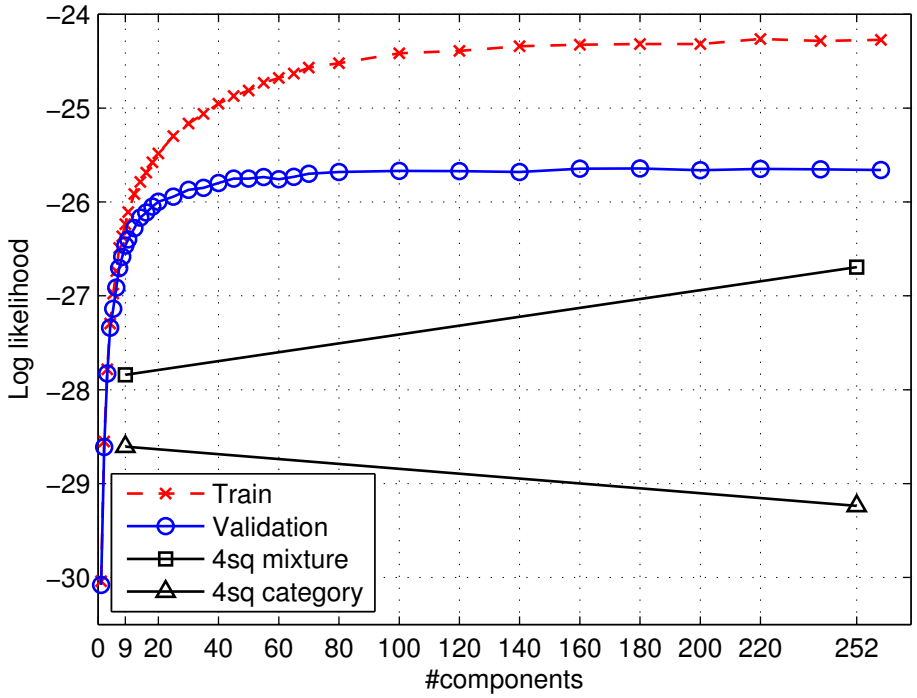

Figure 6.5: Cross-validation likelihoods for different numbers of mixture components. The line with boxes shows the test performance of a mixture model whose latent variables are predefined by the 4sq categories in the training phase. The line with triangles shows the performance of a method that calculates the likelihoods using only the model of each test place's own 4sq category.

curve. After 80 components, the cross-validation likelihood no longer increases significantly and therefore we select 80 as the number of components.

Finally, we train ten mixture models with 80 components using the whole 4sq data set and select the one with the highest training likelihood. The nine components with the highest mixing coefficients of this model are visualized in Figure 6.6. We can observe that the mixture model has captured some place categories that resemble closely certain predefined 4sq categories. For instance, components 4 and 5 could correspond to the school and workplace categories since they are most active in the morning and they are rarely visited on weekends. Component 3 presumably contains some nightlife spots but since the weekly distribution is rather uniform, it probably contains some residences as well. On the other hand, there are some mixture components that are very different from all 4sq categories. For example, we cannot find a clear counterpart for component 6,

which has Sunday as its most active day, among the 4sq categories.

6.4.2 Transfer Coefficient

We compare two strategies for learning the transfer coefficient α (see Equation 6.8). To learn α, we can only use Set A. Therefore, we take Set B as the training data and Set C as the validation data (see Figure 5.1).

The first strategy is to learn a fixed α. We vary α from 0 to 1 in steps of 0.05, and calculate which value gives us the highest prediction accuracy on Set C. The best accuracy is obtained when $\alpha = 0.60$. To learn the varying α, we us Algorithm 6.3.1. The results are shown in Figure 6.7. Looking at the top-left corner of the results table in Figure 6.7 shows that by setting $\alpha(N_p) = 1$ for all N_p gives us a validation accuracy of 53.18 %. By optimizing the alpha values of the nodes one by one starting from $\alpha(N_p = 0)$, we are able to push the test accuracy up to 53.50 %. The right hand side of Figure 6.7 shows the corresponding model.

6.4.3 Next Place Prediction

Now we see if using transfer learning with additional data from 4sq can help us to predict the mobility patterns of the MDC users. Prediction test accuracies are calculated based on Set T (see Figure 5.1) by taking an average of each user's accuracy weighted by the number of test samples the user has. We vary the number of training place transitions per user (N_u) in Set A, starting from the most recent samples, so that we see how the amount of training data affects the performance of transfer learning. If N_a is the user's total number of samples available for training and we calculate the prediction accuracy using N_u training samples, the training indices are the following: $[N_a - N_u + 1, N_a - N_u + 2, \ldots, N_a]$. Test samples always start right after the last training sample and the test data set (Set T) is thus kept fixed. The only exception is that the users who do not have enough training data ($N_a < N_u$) are not taken into account. Thus the number of users decreases when N_u increases, and thus, e.g, there are only ten MDC users with enough data when the number of training samples is 450.

The next place prediction results are shown on the left-hand side of Figure 6.8. The accuracy naturally increases when N_u increases. However, there are also some drops in the accuracy since the set of users on whom the accuracy is calculated might vary. The relative performance of the methods, shown on the right-hand side of Figure 6.8, shows that a 2 percent absolute improvement in prediction accuracy is obtained with transfer learning when N_u is around 20. After 60 training samples for the

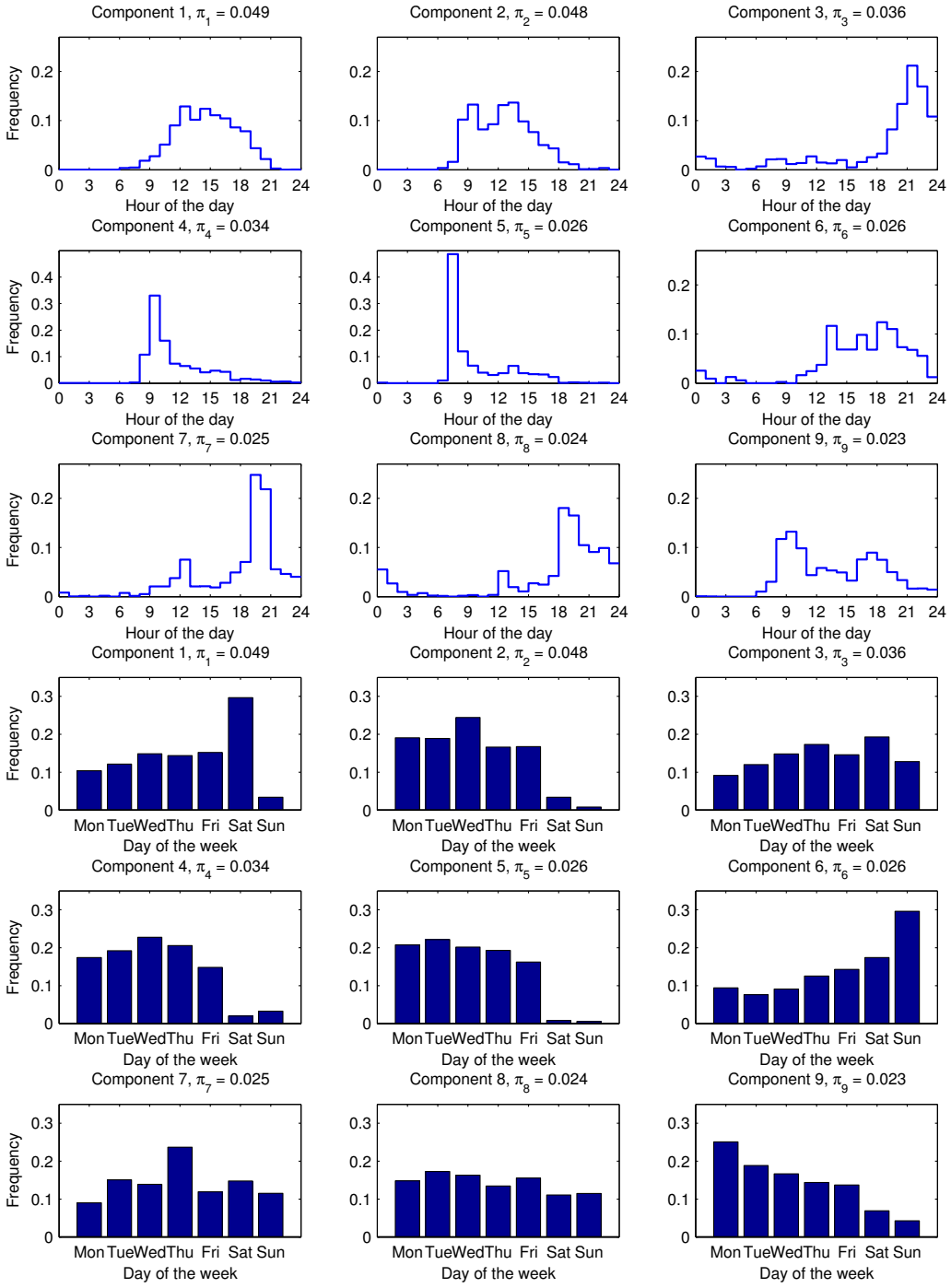

Figure 6.6: The daily and weekly check-in time distributions of the nine components with the highest mixing coefficients in the 4sq mixture model.

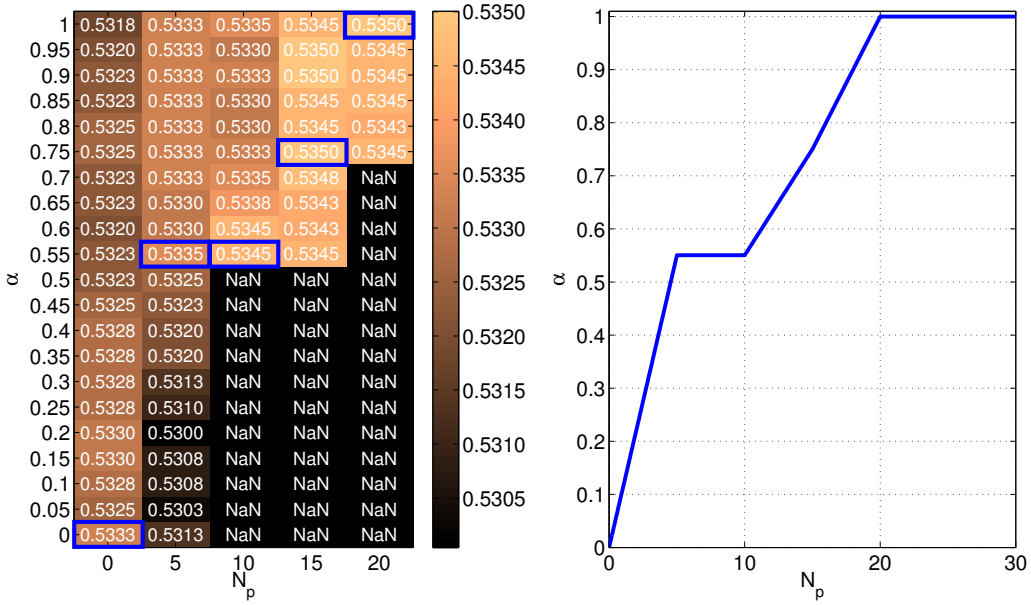

Figure 6.7: Optimization of the varying alpha model. Left: intermediate results with optimal alphas for each column marked by blue rectangles. Right: the resulting piecewise function $\alpha(N_p)$. N_p is is the number of previous visits to the MDC place and a NaN value means that the accuracy for the corresponding value of the node has not been calculated since the function is assumed to be monotonically increasing.

fixed α model and 160 training samples for the varying α model, transfer learning no longer improves the prediction accuracy but makes it about 0.5 percentage points lower than the baseline performance of the MDC only model. We suspect that the reason for this is that when there is a sufficient amount of training data, the time distributions of the most common places can be estimated accurately using only the previous MDC data. At this point, the PPD used for transfer learning may only make the time distributions of some rarely visited places too "peaky" so that they can get predicted even though they have been visited very rarely in the past. Of the two transfer learning methods, the fixed α model seems to perform better than the varying α model when N_u is below 180. This suggests that the varying α model does not have as good generalization performance as the fixed α method.

Transfer learning is only applied to the time distributions but not to the Markov model part of Equation 5.9 which mainly dominates the predictions. To suppress the impact of the Markov model and understand the sole effect of transfer learning, we calculate the prediction accuracies using

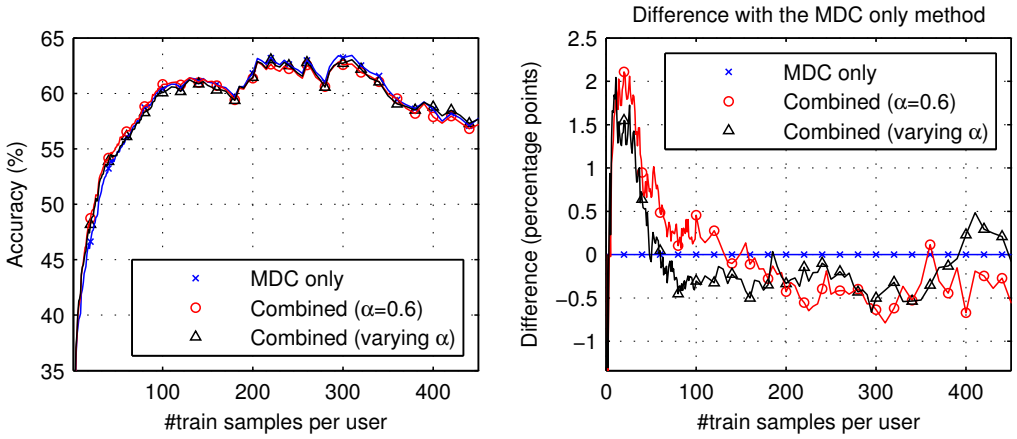

Figure 6.8: Left: next place prediction accuracy as a function of the number of place transitions (N_u) used for training. The predictions are done with three variants of our approach (see Section 5.3): without transfer learning (MDC only), using the fixed α transfer learning method, and using the varying α transfer learning method. Right: accuracy differences between the MDC only method and the transfer learning methods.

a 0-order Markov model instead of the standard 1st-order Markov model. The results are shown in Figure 6.9. Now we can see the difference more clearly: transfer learning brings a 4 percent absolute improvement when N_u is around 20. Furthermore, transfer learning can perform comparably to the MDC only model even after 160 training samples.

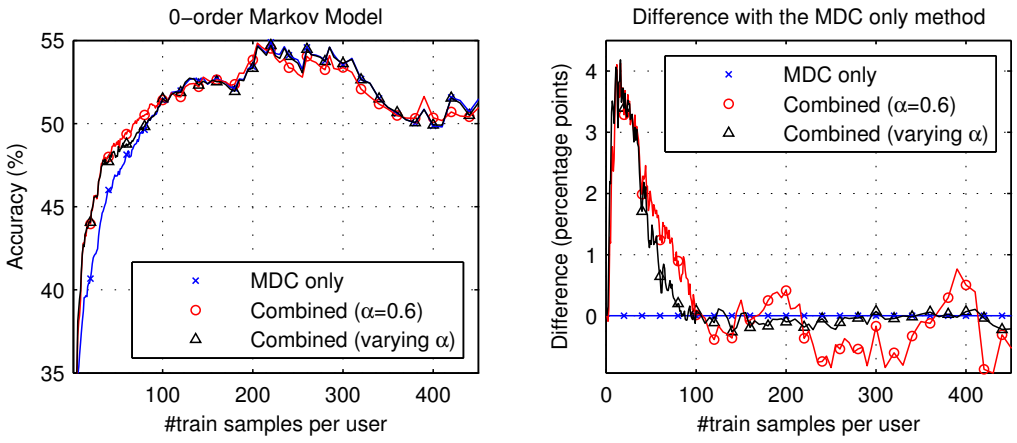

Figure 6.9: The same results as in Figure 6.8 but now the Markov model part in Equation 5.9 has been replaced by a 0-order Markov model.

In order to understand the relation between N_u and the length of the

data collection period, we calculate the number of days during which a user has made the N_u visits. These time periods vary over users since some people are more active than others so we calculate the average number of days over users. We have seen that transfer learning provides the largest improvements when N_u is around 20, which corresponds to 9 days. On the other hand, we saw that the two transfer learning methods improve the predictions up to 60 and 160 training samples. The average lengths of the corresponding time periods are 27 and 70 days, respectively. Thus, we may conclude that the next place prediction accuracy can be improved during the first month of data collection by including additional data from 4sq.

7

Discussion and Conclusions

In this thesis, we have conducted a large-scale study on human mobility prediction using data from smartphone sensors (Mobile Data Challenge) and a location-based social network (Foursquare). The main goal has been to develop a probabilistic next place prediction method which uses the two data sets in a complementary manner. To our knowledge, this is the first time that transfer learning has been used for analyzing mobility data.

Our first goal was to study the differences and similarities of the two data sets. The nature of the two location disclosure systems is very different. While the visits in MDC are recorded automatically, the 4sq data set consists of manual check-ins. This difference is reflected in the amount and the type of produced check-ins as 4sq users make less check-ins on average and they have a stronger preference on checking in to new places. The daily and weekly check-in time distributions were found to be surprisingly similar for the two data sets suggesting that useful knowledge about the temporal aspect of check-ins could be transferred from one data set to another. Furthermore, the ranks of the transitions in the data sets followed the same distribution, which means that it might be possible to transfer also knowledge about the transitional aspects of check-in sequences. Nevertheless, in this work we focused on the temporal domain. We also studied the problem of matching places across data sets in order to facilitate transfer learning. The matching worked relatively well for certain types of places but we eventually adopted another way of transferring relevant knowledge due to low general matching accuracy.

The second goal was to derive a probabilistic next place prediction model which is applicable to transfer learning. We identified that all of the three best methods in the MDC competition consist of a transitional part (namely a Markov model) and a temporal part, and we implemented method M1, which finished 3rd in the competition, and method M2, which

is the probabilistic part of the winning approach. In our experiments, M1 outperformed M2 suggesting that the secret behind the winning approach was not in M2 itself but in the fact that it used an ensemble of M2 and two discriminative classifiers and that it implemented some minor improvements such as the detection of a change in residential location. We showed that M1 can be further improved by using multinomial distributions or kernel density estimation for the time distributions instead of Gaussians. Then we derived our approach based on M1. This approach additionally incorporates an idea of summing over the possible transition times, which was introduced in M2, and in consequence the model does not use the end time distributions but the start time distributions of the visits of a place. This means that we can utilize the 4sq data since instead of the end times of the visits it contains the check-in times which roughly correspond to the start times of the visits. In addition to allowing transfer learning, we showed that our approach outperforms both M1 and M2 in their simplest formulations. The prediction accuracies of M1, M2, and our approach were 51.6 %, 50.2 %, and 52.6 %, respectively.

Our third goal was to use 4sq data to improve the predictions for the MDC users. To achieve this goal, we first clustered 4sq places and learned the time distributions of the clusters in an unsupervised manner using a mixture of multinomials. Then we learned the time distribution of each MDC place given the previous visits to the place and the 4sq clusters. This approach is based on the posterior predictive distribution and it estimates how well the previous MDC visit times fit to different 4sq clusters and then learns the time distribution, not directly based on the previous visits, but by taking a weighted average of the cluster distributions. We showed that this kind of a transfer learning approach increases the prediction accuracy on average by up to 2 percentage points during the first month of data collection. This is an important result since it tackles the *cold start problem* which is encountered in many modeling tasks at the initial phase when there is little data making the estimation of any distributions challenging.

We conclude by discussing some areas for future work. First of all, we noticed that smoothing plays an important role when learning the time distributions of places since most of the places have only a few prior visits. Having tried out few different strategies, a simple Laplace smoothing proved to provide good results but a systematic study of different smoothing techniques is called for. Applying different smoothing strategies for the transitional models could also improve the predictions. Secondly, in this work, transfer learning was used to learn time distributions by utilizing another data set, but there are also other ways how transfer learning could be applied to the next place prediction problem. For instance,

the approach introduced in this work is directly applicable to transferring knowledge, not only from other data sets, but also from other users in the same data set. Furthermore, applying transfer learning, not only for the temporal models, but also for the transitional models would be an interesting future direction.

Bibliography

[1] E. Alpaydin. *Introduction to Machine Learning*. MIT press, 2004.

[2] J. Biesterfeld, E. Ennigrou, and K. Jobmann. Location prediction in mobile networks with neural networks. In *Proceedings of the International Workshop on Applications of Neural Networks to Telecommunications*, volume 97, pages 207–214, 1997.

[3] C.M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[4] Z. Cheng, J. Caverlee, K. Lee, and D.Z. Sui. Exploring millions of footprints in location sharing services. In *Proceedings of the 5th International AAAI Conference on Weblogs and Social Media*, 2011.

[5] E. Cho, S.A. Myers, and J. Leskovec. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1082–1090, 2011.

[6] H. Cramer, M. Rost, and L.E. Holmquist. Performing a check-in: Emerging practices, norms and 'conflicts' in location-sharing using Foursquare. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*, pages 57–66, 2011.

[7] S.K. Das and S.K. Sen. Adaptive location prediction strategies based on a hierarchical network model in a cellular mobile environment. *The Computer Journal*, 42(6):473–486, 1999.

[8] M. De Domenico, A. Lima, and M. Musolesi. Interdependence and predictability of human mobility and social interactions. In *Proceedings of the Mobile Data Challenge by Nokia Workshop in conjunction with International Conference on Pervasive Computing*, 2012.

[9] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38, 1977.

[10] W. Dong, K. Heller, and A. Pentland. Modeling infection with multi-agent dynamics. In *Proceedings of the International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction*, pages 172–179. Springer, 2012.

[11] N. Eagle and A. Pentland. Reality mining: Sensing complex social systems. *Personal and Ubiquitous Computing*, 10(4):255–268, 2006.

[12] N. Eagle and A.S. Pentland. Eigenbehaviors: Identifying structure in routine. *Behavioral Ecology and Sociobiology*, 63(7):1057–1066, 2009.

[13] V. Etter, M. Kafsi, and E. Kazemi. Been there, done that: What your mobility traces reveal about your behavior. In *Proceedings of the Mobile Data Challenge by Nokia Workshop in conjunction with International Conference on Pervasive Computing*, 2012.

[14] H. Gao, J. Tang, and H. Liu. Exploring social-historical ties on location-based social networks. In *Proceedings of the 6th International AAAI Conference on Weblogs and Social Media*, 2012.

[15] H. Gao, J. Tang, and H. Liu. Mobile location prediction in spatio-temporal context. In *Proceedings of the Mobile Data Challenge by Nokia Workshop in conjunction with International Conference on Pervasive Computing*, 2012.

[16] M.C. Gonzalez, C.A. Hidalgo, and A.L. Barabási. Understanding individual human mobility patterns. *Nature*, 453(7196):779–782, 2008.

[17] J. Hightower, S. Consolvo, A. LaMarca, I. Smith, and J. Hughes. Learning and recognizing the places we go. *UbiComp 2005: Ubiquitous Computing*, pages 903–903, 2005.

[18] J.H. Kang, W. Welbourne, B. Stewart, and G. Borriello. Extracting places from traces of locations. In *Proceedings of the 2nd ACM international workshop on Wireless mobile applications and services on WLAN hotspots*, pages 110–118, 2004.

[19] M. Kim, D. Kotz, and S. Kim. Extracting a mobility model from real user traces. In *Proceedings of the 25th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 1–13, 2006.

[20] N. Kiukkonen, J. Blom, O. Dousse, D. Gatica-Perez, and J. Laurila. Towards rich mobile phone datasets: Lausanne data collection campaign. In *Proceedings of the 7th International Conference on Pervasive Services*, 2010.

[21] Y. Koren. The BellKor solution to the Netflix grand prize. *Netflix Prize Documentation*, 2009.

[22] S. Kullback and R.A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951.

[23] A. LaMarca, Y. Chawathe, S. Consolvo, J. Hightower, I. Smith, J. Scott, T. Sohn, J. Howard, J. Hughes, F. Potter, et al. Place lab: Device positioning using radio beacons in the wild. *Pervasive Computing*, pages 301–306, 2005.

[24] J. Laurila, D. Gatica-Perez, I. Aad, J. Blom, O. Bornet, T.-M.-T. Do, O. Dousse, J. Eberle, and M. Miettinen. The mobile data challenge: Big data for mobile computing research. In *Proceedings of the Mobile Data Challenge by Nokia Workshop, in conjunction with International Conference on Pervasive Computing*, 2012.

[25] J. Lindqvist, J. Cranshaw, J. Wiese, J. Hong, and J. Zimmerman. I'm the mayor of my house: Examining why people use Foursquare – a social-driven location sharing application. In *Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems*, pages 2409–2418, 2011.

[26] E. Malmi, T.M.T. Do, and D. Gatica-Perez. Checking in or checked in: Comparing large-scale manual and automatic location disclosure patterns. In *Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia*, 2012.

[27] C.D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. MIT press, 1999.

[28] G.J. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. Wiley-Interscience, 2007.

[29] R. Montoliu and D. Gatica-Perez. Discovering human places of interest from multimodal mobile phone data. In *Proceedings of the 9th International Conference on Mobile and Ubiquitous Multimedia*, 2010.

[30] A. Noulas, S. Scellato, R. Lambiotte, M. Pontil, and C. Mascolo. A tale of many cities: Universal patterns in human urban mobility. *PLoS ONE*, 7:e37027, May 2012.

[31] A. Noulas, S. Scellato, N. Lathia, and C. Mascolo. Mining user mobility features for next place prediction in location-based services. In *Proceedings of the IEEE Internationcal Conference on Data Mining*, pages 1038–1043, 2012.

[32] P. Nurmi and S. Bhattacharya. Identifying meaningful places: The non-parametric way. In *Proceedings of the 6th International Conference on Pervasive Computing*, pages 111–127, 2008.

[33] S.J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

[34] L. Rigouste, O. Cappé, and F. Yvon. Inference and evaluation of the multinomial mixture model for text clustering. *Information processing & management*, 43(5):1260–1280, 2007.

[35] C. Song, Z. Qu, N. Blumm, and A.L. Barabási. Limits of predictability in human mobility. *Science*, 327(5968):1018–1021, 2010.

[36] K.P. Tang, J. Lin, J.I. Hong, D.P. Siewiorek, and N. Sadeh. Rethinking location sharing: exploring the implications of social-driven vs. purpose-driven location sharing. In *Proceedings of the 12th ACM International Conference on Ubiquitous Computing*, pages 85–94, 2010.

[37] Y.W. Teh. A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 985–992, 2006.

[38] A. Töscher, M. Jahrer, and R.M. Bell. The BigChaos solution to the Netflix grand prize. *Netflix Prize Documentation*, 2009.

[39] J. Wang and B. Prabhala. Periodicity based next place prediction. In *Proceedings of the Mobile Data Challenge by Nokia Workshop in conjunction with International Conference on Pervasive Computing*, 2012.

[40] M. Ye, K. Janowicz, C. Mülligann, and W.C. Lee. What you are is when you are: The temporal dimension of feature types in location-based social networks. In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 102–111, 2011.

[41] V.W. Zheng, Y. Zheng, X. Xie, and Q. Yang. Collaborative location and activity recommendations with GPS history data. In *Proceedings of the 19th International Conference on World Wide Web*, pages 1029–1038, 2010.

Appendix A

Derivation of the EM Equations

In this appendix, we first present how to evaluate the log likelihood function so that we avoid underflow and then we derive the *expectation* and *maximization* steps of the EM algorithm for our mixture of multinomials. Full descriptions of the parameters appearing in this chapter can be found in Section 6.2. Note that our mixture model slightly differs from a standard multinomial mixture since each component, in our case, contains two distributions (the daily and weekly check-in time distributions) rather than only a single multinomial. Nevertheless, this has little effect on the EM equations as it turns out.

A.1 Avoiding Underflow

As given in Equation 6.2, the log likelihood function of the check-in times of all places $\mathbf{T}$ is given by

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\varphi} \mid \mathbf{T}) = \sum_{l=1}^{L} \left[\log(A_l) + \log \sum_{c=1}^{C} \pi_c \left(\prod_{i=0}^{23} \theta_{ci}^{\mathcal{H}_{li}} \right) \left(\prod_{j=1}^{7} \varphi_{cj}^{\mathcal{D}_{lj}} \right) \right].$$

The problem with this formula is that when we have lots of check-in times it becomes numerically unstable due to the product of many small terms. In order to avoid underflow, let us do some modifications to the log likeli-

hood function

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\varphi} \mid \mathbf{T}) &= \sum_{l=1}^{L}\left[\log(A_l) + \log\sum_{c=1}^{C}\exp\left(\log\left(\pi_c\left(\prod_{i=0}^{23}\theta_{ci}^{\mathcal{H}_{li}}\right)\left(\prod_{j=1}^{7}\varphi_{cj}^{\mathcal{D}_{lj}}\right)\right)\right)\right] \\
&= \sum_{l=1}^{L}\left[\log(A_l) + \log\sum_{c=1}^{C}\exp\Bigg(\log(\pi_c) + \sum_{i=0}^{23}\mathcal{H}_{li}\log\theta_{ci} \right. \\
&\quad \left. + \sum_{j=1}^{7}\mathcal{D}_{lj}\log\varphi_{cj}\Bigg)\right] \\
&= \sum_{l=1}^{L}\left[\log(A_l) + \log\sum_{c=1}^{C}\exp\Bigg(B_l - B_l + \log(\pi_c) + \sum_{i=0}^{23}\mathcal{H}_{li}\log\theta_{ci} \right. \\
&\quad \left. + \sum_{j=1}^{7}\mathcal{D}_{lj}\log\varphi_{cj}\Bigg)\right] \\
&= \sum_{l=1}^{L}\left[\log(A_l) + B_l + \log\sum_{c=1}^{C}\exp\Bigg(-B_l + \log(\pi_c) + \sum_{i=0}^{23}\mathcal{H}_{li}\log\theta_{ci} \right. \\
&\quad \left. + \sum_{j=1}^{7}\mathcal{D}_{lj}\log\varphi_{cj}\Bigg)\right],
\end{aligned}$$

where

$$B_l = \max_c\left\{\log(\pi_c) + \sum_{i=0}^{23}\mathcal{H}_{li}\log\theta_{ci} + \sum_{j=1}^{7}\mathcal{D}_{lj}\log\varphi_{cj}\right\}.$$

This is called the *log-sum-exp trick*[1] .

A.2 E Step

In the expectation step, we need to evaluate

$$\mathcal{Q}(\boldsymbol{\Theta}, \boldsymbol{\Theta}^{\text{old}}) = \sum_{\mathbf{Z}} p(\mathbf{Z} \mid \mathbf{T}, \boldsymbol{\Theta}^{\text{old}})\log p(\mathbf{T}, \mathbf{Z} \mid \boldsymbol{\Theta}), \tag{A.1}$$

where $\boldsymbol{\Theta} = \{\boldsymbol{\theta}, \boldsymbol{\varphi}\}$ are the mixture parameters and $\mathbf{T} = [\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_L]$ are the check-in time vectors of each place. We also use another representation

[1] `http://machineintelligence.tumblr.com/post/4998477107/the-log-sum-exp-trick`

$\mathbf{T} = (\mathcal{H}, \mathcal{D})$, where $\mathcal{H}_{li}$ and $\mathcal{D}_{lj}$ are the numbers of check-ins to place l falling into ith hour and jth weekday, respectively.

Let us first calculate the posterior probability of a single latent variable z_{lc}, which is a 0/1 hidden variable indicating whether the check-in times of place l were generated by component c

$$\begin{aligned} p(z_{lc} = 1 \mid \mathbf{t}_l, \boldsymbol{\theta}, \boldsymbol{\varphi}) &= \frac{p(z_{lc} = 1 \mid \boldsymbol{\theta}, \boldsymbol{\varphi}) p(\mathbf{t}_l \mid z_{lc} = 1, \boldsymbol{\theta}, \boldsymbol{\varphi})}{p(\mathbf{t}_l \mid \boldsymbol{\theta}, \boldsymbol{\varphi})} \\ &= \frac{\pi_c \left(\prod_{i=0}^{23} \theta_{ci}^{\mathcal{H}_{li}}\right) \left(\prod_{j=1}^{7} \varphi_{cj}^{\mathcal{D}_{lj}}\right)}{\sum_{c'=1}^{C} \pi_{c'} \left(\prod_{i=0}^{23} \theta_{c'i}^{\mathcal{H}_{li}}\right) \left(\prod_{j=1}^{7} \varphi_{c'j}^{\mathcal{D}_{lj}}\right)} \\ &= \exp\Bigg(\log(\pi_c) + \sum_{i=0}^{23} \mathcal{H}_{li} \log \theta_{ci} + \sum_{j=1}^{7} \mathcal{D}_{lj} \log \varphi_{cj} \\ &\quad - \log \sum_{c'=1}^{C} \pi_{c'} \left(\prod_{i=0}^{23} \theta_{c'i}^{\mathcal{H}_{li}}\right) \left(\prod_{j=1}^{7} \varphi_{c'j}^{\mathcal{D}_{lj}}\right) \Bigg) \equiv \gamma_{lc}. \end{aligned} \tag{A.2}$$

The logarithmic term, that is subtracted from the rest, is similar to $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\varphi} \mid \mathbf{T})$, so we can again apply the log-sum-exp trick to the above expression.

Then, in order to calculate the complete-data log likelihood $\log p(\mathbf{T}, \mathbf{Z} \mid \boldsymbol{\Theta})$, we need the following

$$\begin{aligned} \log p(\mathbf{T} \mid \mathbf{Z}, \boldsymbol{\Theta}) &= \log \prod_{l=1}^{L} p(\mathbf{t}_l \mid \mathbf{z}_l, \boldsymbol{\Theta}) \\ &= \log \prod_{l=1}^{L} \prod_{c=1}^{C} p(\mathbf{t}_l \mid \boldsymbol{\theta}_c, \boldsymbol{\varphi}_c)^{z_{lc}} \\ &= \sum_{l=1}^{L} \sum_{c=1}^{C} \log p(\mathbf{t}_l \mid \boldsymbol{\theta}_c, \boldsymbol{\varphi}_c)^{z_{lc}}, \end{aligned}$$

and also

$$\log p(\mathbf{Z} \mid \boldsymbol{\Theta}) = \sum_{l=1}^{L} \sum_{c=1}^{C} \log \pi_c^{z_{lc}}.$$

Now we can write

$$\begin{aligned}\log p(\mathbf{T}, \mathbf{Z} \mid \mathbf{\Theta}) &= \log p(\mathbf{T} \mid \mathbf{Z}, \mathbf{\Theta}) + \log p(\mathbf{Z} \mid \mathbf{\Theta}) \\ &= \sum_{l=1}^{L} \sum_{c=1}^{C} \log p(\mathbf{t}_l \mid \boldsymbol{\theta}_c, \boldsymbol{\varphi}_c)^{z_{lc}} + \log \pi_c^{z_{lc}} \\ &= \sum_{l=1}^{L} \sum_{c=1}^{C} z_{lc} \Bigg[\log \frac{N_l!}{\prod_{i=0}^{23} \mathcal{H}_{li}!} + \sum_{i=0}^{23} \mathcal{H}_{li} \log \theta_{ci} \\ &\quad + \log \frac{N_l!}{\prod_{j=1}^{7} \mathcal{D}_{lj}!} + \sum_{j=1}^{7} \mathcal{D}_{lj} \log \varphi_{cj} + \log \pi_c \Bigg]. \end{aligned} \tag{A.3}$$

Finally, plugging Equations A.2 and A.3 into A.1, we obtain

$$\begin{aligned}\mathcal{Q}(\mathbf{\Theta}, \mathbf{\Theta}^{\text{old}}) = \sum_{l=1}^{L} \sum_{c=1}^{C} \gamma_{lc} \Bigg[&\log \frac{N_l!}{\prod_{i=0}^{23} \mathcal{H}_{li}!} + \sum_{i=0}^{23} \mathcal{H}_{li} \log \theta_{ci} \\ &+ \log \frac{N_l!}{\prod_{j=1}^{7} \mathcal{D}_{lj}!} + \sum_{j=1}^{7} \mathcal{D}_{lj} \log \varphi_{cj} + \log \pi_c \Bigg]. \end{aligned} \tag{A.4}$$

A.3 M Step

From Equations 3.2 and A.4, we can derive the following optimization problem

$$\begin{aligned}\underset{\theta_{ci}, \varphi_{cj}, \pi_c}{\arg\max} \sum_{l=1}^{L} \sum_{c=1}^{C} \gamma_{lc} \Bigg[&\log \frac{N_l!}{\prod_{i=0}^{23} \mathcal{H}_{li}!} + \sum_{i=0}^{23} \mathcal{H}_{li} \log \theta_{ci} \\ &+ \log \frac{N_l!}{\prod_{j=1}^{7} \mathcal{D}_{lj}!} + \sum_{j=1}^{7} \mathcal{D}_{lj} \log \varphi_{cj} + \log \pi_c \Bigg] \\ \text{subject to} \sum_{i=0}^{23} \theta_{ci} = 1, \quad & c = 1, 2, \ldots, C \\ \sum_{j=1}^{7} \varphi_{cj} = 1, \quad & c = 1, 2, \ldots, C \\ \sum_{c=1}^{C} \pi_c = 1. & \end{aligned}$$

Due to the equality constraints, we incorporate the technique of *Lagrange multipliers* to solve the optimization task. The *Lagrangian* takes the form

$$F(\boldsymbol{\theta}, \boldsymbol{\varphi}, \boldsymbol{\pi}, \boldsymbol{\lambda}) = \mathcal{Q}(\boldsymbol{\Theta}, \boldsymbol{\Theta}^{\text{old}}) + \sum_{c=1}^{C} \left[\lambda_{c1} \left(\sum_{i=0}^{23} \theta_{ci} - 1 \right) + \lambda_{c2} \left(\sum_{j=1}^{7} \varphi_{cj} - 1 \right) \right] + \lambda_3 \left(\sum_{c=1}^{C} \pi_c - 1 \right).$$

Now, by solving $\nabla F = 0$, we obtain the following update equations for the parameters

$$\theta_{ci}^{\text{new}} = \frac{\sum_{l=1}^{L} \gamma_{lc} \mathcal{H}_{li}}{L_c}, \quad \varphi_{cj}^{\text{new}} = \frac{\sum_{l=1}^{L} \gamma_{lc} \mathcal{D}_{lj}}{L_c}, \quad \pi_c = \frac{L_c}{L},$$

where $L_c = \sum_{l=1}^{L} \gamma_{lc}$.

Finally, we apply additive smoothing to parameters θ and φ as suggested by Rigouste et al. [34]. We use the smoothing value of 0.1 that was found to give optimal classification results for a text document classification task in [34]. The resulting update equations are the following

$$\theta_{ci}^{\text{new}} = \frac{0.1 + \sum_{l=1}^{L} \gamma_{lc} \mathcal{H}_{li}}{2.4 + L_c}, \quad i = 1, 2, \ldots, 24 \tag{A.5}$$

$$\varphi_{cj}^{\text{new}} = \frac{0.1 + \sum_{l=1}^{L} \gamma_{lc} \mathcal{D}_{lj}}{0.7 + L_c}, \quad j = 1, 2, \ldots, 7 \tag{A.6}$$

$$\pi_c = \frac{L_c}{L}. \tag{A.7}$$

The constants 2.4 and 0.7 in the numerators of θ_{ci}^{new} and $\varphi_{cj}^{\text{new}}$ have been selected so that the probabilities sum up to one.